UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

Concepção e Implementação de um Agente Semiótico como Parte de um Modelo Social de Aprendizagem a Distância

por

JOÃO LUIZ JUNG

Dissertação submetida à avaliação,
como requisito parcial para a obtenção do grau de Mestre
em Ciência da Computação

Profa. Dra. Rosa Maria Vicari
Orientadora

Prof. Dr. Rafael Heitor Bordini
Co-orientador

Porto Alegre, dezembro de 2001.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Jung, João Luiz

Concepção e Implementação de um Agente Semiótico como Parte de um Modelo Social de Aprendizagem a Distância / por João Luiz Jung. – Porto Alegre: PPGC da UFRGS, 2001.

98 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR – RS, 2001. Orientadora: Vicari, Rosa Maria; Co-orientador: Bordini, Rafael Heitor.

1. Sistemas multiagentes. 2. Sistemas tutores inteligentes. 3. Educação a distância. 4. Teorias sócio-interacionistas. 5. Engenharia semiótica. I. Vicari, Rosa Maria. II. Bordini, Rafael Heitor. III. Título.

Agradecimentos

Agradeço aos meus colegas do Curso de Pós-Graduação em Ciência da Computação da UFRGS, Mestrado e Doutorado, pelo prazer de suas amizades, conversas e trocas de conhecimentos, em especial às minhas colegas doutorandas Adja Ferreira de Andrade e Patrícia Augustin Jaques que prestaram inestimável apoio.

A todos os professores do Curso de Pós-Graduação em Ciência da Computação da UFRGS com quem tive a oportunidade de conviver e adquirir conhecimento; a todos os funcionários e colaboradores do Instituto de Informática da UFRGS que não medem esforços para o cumprimento de sua missão.

Ao 1º Centro de Telemática de Área, do Ministério da Defesa, Exército Brasileiro, em especial ao TenCel Elifas Chaves Gurgel do Amaral, chefe do 1º CTA, pelo apoio e incentivo aos seus subordinados em adquirir novos conhecimentos, oportunizando assim o desenvolvimento desta dissertação.

Ao professor Marcelo Pimenta pelos momentos de incentivo e por ter auxiliado nas referências de Engenharia Semiótica.

Ao meu co-orientador Rafael Heitor Bordini pelas contribuições, sugestões e trocas de idéias ao longo do trabalho.

À minha orientadora Rosa Maria Vicari pelo incentivo, apoio, dedicação e carinho nestes anos de convivência para o desenvolvimento deste trabalho.

Aos meus pais Luiz Frederico Jung e Lizete Suzel Jung e a toda a minha família pelo carinho, compreensão e apoio irrestrito em todos os sentidos.

Finalmente, à minha esposa Cacilda Freitas Jung pelo apoio, compreensão, estímulo, confiança e principalmente pelo entendimento das muitas horas de dedicação, que soube aceitar a minha ausência nas horas de lazer, e, mesmo assim, incentivou para que eu pudesse realizar este sonho.

"*Ele não sabia que era impossível. Foi lá e fez.*" Jean Cocteau

Sumário

Lista de Abreviaturas

ACL *Agent Communication Language*
ANS Agente Servidor de Nomes
API *Application Program Interface*
ARPA *Advanced Research Projects Agency*
AUT autônomo
BCV Base de Conhecimento Virtual
BDI *belief – desire - intention*
CAI *Computer Assisted Instruction*
CTA Centro de Telemática de Área
DAI *Distributed Artificial Intelligence*
DEP dependente
DPS *Distributed Problem Solving*
DTC *Design-To-Criteria*
EAD Educação a Distância
FAQ perguntas mais freqüentes
GIA Grupo de Inteligência Artificial
HTML *HyperText Markup Language*
HTTP *HyperText Transfer Protocol*
IA Inteligência Artificial
IAD Inteligência Artificial Distribuída
ICAI *Intelligent Computer Assisted Instruction*
IHC Interação Homem-Computador
IND independência
ITS *Intelligent Tutoring Systems*
JADE *Java Agent framework for Distance learning Environments*
JAT *Java Agent Template*
JDBC *Java Data Base Connection*
KIF *Knowledge Interchange Format*
KQML *Knowledge Query and Manipulation Language*
KRSS *Knowledge Representation System Specification*
KSE *Knowledge Sharing Effort*
LBMD *Locally Believed Mutual Dependence*
LBRD *Locally Believed Reciprocal Dependence*
LEMD Linguagem de Especificação da Mensagem do *Designer*
MBMD *Mutually Believed Mutual Dependence*
MBRD *Mutually Believed Reciprocal Dependence*
MCOE *Multi-agent Co-operative Environment*
MI *Medium Interface*
NDP Nível de Desenvolvimento Potencial
NDR Nível de Desenvolvimento Real
NG *no goal*
NN *Not Null*
NP *no plans*
OO Orientação a Objetos
PK *Primary Key*

PRS	*Procedural Reasoning System*
RMI	*Remote Method Invocation*
SEM	Sociedade dos Estados Mentais
SI	Signo de Interface
SMA	Sistemas Multiagentes
SQL	*Structured Query Language*
STI	Sistemas Tutores Inteligentes
TCP	*Transmission Control Protocol*
UD	*Unilateral Dependence*
VKB	*Virtual Knowledge Base*
WEB	*World Wide Web*
XML	*eXtensible Markup Language*
XSL	*eXtensible Style Sheets*
ZDP	Zona de Desenvolvimento Proximal

Lista de Figuras

Lista de Tabelas

Resumo

Esta dissertação situa-se no projeto de pesquisa intitulado "Um Modelo Computacional de Aprendizagem a Distância Baseada na Concepção Sócio-Interacionista". Este projeto se enquadra na visão de aprendizagem situada, isto é, na concepção de cognição como uma prática social baseada na utilização de linguagem, símbolos e signos. O objetivo é a construção de um ambiente de Educação a Distância, implementado como um sistema multiagente composto por agentes artificiais e agentes humanos, inspirando-se na teoria sócio-interacionista de Vygotsky. Nesta sociedade, todos os personagens (aprendizes e agentes artificiais) são modelados como agentes sociais integrados em um ambiente de ensino-aprendizagem. A arquitetura deste sistema é formada pelos seguintes agentes artificiais: agente diagnóstico, agente mediador, agente colaborativo, agente semiótico e agente social. Os agentes humanos que interagem com o sistema desempenham o papel de tutores, aprendizes ou ambos. Esta dissertação visa à concepção e à implementação de um dos agentes desta arquitetura: o agente semiótico. Esta concepção foi baseada na Engenharia Semiótica, em particular para a apresentação do material instrucional utilizado no processo de ensino-aprendizagem.

Palavras-chave: Sistemas Multiagentes, Sistemas Tutores Inteligentes, Educação a Distância, Teorias Sócio-Interacionistas, Engenharia Semiótica.

TITLE: “CONCEPTION AND IMPLEMENTATION OF A SEMIOTIC AGENT AS PART OF A SOCIAL MODEL OF DISTANCE LEARNING”

Abstract

This dissertation is part of a research project called "A Computational Model of Distance Learning based on the Socio-Interactionist Approach". This project is related to situated learning, i.e., in the conception of cognition as a social practice based on the use of language, symbols and signs. The objective is the construction of a Distance Learning environment, implemented as a multiagent system composed of artificial and human agents, and inspired by Vygotsky's socio-interactionist theory. In this society, all characters (pupils and artificial agents) are modelled as social agents integrated in a teaching-learning environment. The architecture of this system is formed by the following artificial agents: diagnostic agent, mediator agent, collaborative agent, semiotic agent, and social agent. The human agents who interact with the sistem play the role of tutor, pupils, or both. This dissertation aims at the conception and the implementation of one of the agents from such architecture: the semiotic one. This conception was based on Semiotic Engineering, in particular for the presentation of the instructional material used in the teaching-learning process.

Keywords: Multi-Agent Systems, Intelligent Tutoring Systems, Distance Education, Socio-Interactionist Pedagogical Theories, Semiotic Engineering.

1 Introdução

A aprendizagem é um processo de construção e interação que não pode mais ser vista dentro de um modelo instrucionista, behaviorista ou de auto-estudo. Com o advento das redes de computadores e a multimídia, surge uma nova modalidade de ensino-aprendizagem: a aprendizagem colaborativa. No entanto, em tal modalidade, pouca atenção tem sido dada à afetividade e à motivação do aprendiz no processo de interação e cada vez mais os ambientes computacionais têm reproduzido o modelo competitivo da sala de aula tradicional, priorizando muitas vezes o estudo individualizado e o instrucionismo.

É um desafio dos profissionais de informática educativa resgatar esta mudança de foco. É necessário favorecer um processo educativo que fomente a intensa participação interativa e colaborativa dos alunos, em outras palavras, encontrar um modelo de Educação a Distância (EAD) que privilegie o coletivo e o social e não apenas o indivíduo.

O nosso trabalho visa a concepção de um agente semiótico inserido em um ambiente educacional modelado como parte da arquitetura multiagente do projeto "Um Modelo Computacional de Aprendizagem a Distância Baseada na Concepção Sócio-Interacionista", inspirado nas idéias de Vygotsky [VYG 98] e [VYG 98a], Levy [LEV 99], Freire & Fagundes [FRE 95], Clermont & Nelly [CLE 78] e Baquero [BAQ 98], ora em desenvolvimento, onde se encontram em andamento duas teses de doutorado e duas dissertações de mestrado, dentre as quais esta faz parte.

Esta pesquisa faz uso da tecnologia de Inteligência Artificial Distribuída (IAD), especificamente Sistema Multiagente (SMA), para implementar computacionalmente este modelo social para aprendizagem a distância. O sistema está subdividido em cinco agentes artificiais que irão dar suporte à aprendizagem a distância: *agente diagnóstico* que implementa o conceito de zona de desenvolvimento proximal defendida por Vygotsky, visando transformar habilidades potenciais em habilidades reais, buscando expandir a capacidade de desenvolvimento sócio-cognitiva do aluno; o *agente mediador* que auxilia no processo de internalização do aluno decorrente do contato com o ambiente social de EAD; o *agente colaborativo* que é responsável por mediar/monitorar a interação entre grupos de alunos em ferramentas síncronas de comunicação entre os alunos (por exemplo, *chat*); o *agente social* que deve estabelecer a integração da sociedade formando grupos de alunos para estudo e criando um agente colaborativo para cada grupo formado; e o **agente semiótico** responsável pela utilização de signos, conceitos e linguagem repassados ao agente mediador ou agente colaborativo e, conseqüentemente, ao aluno no processo de aprendizagem.

Esta dissertação tem como objetivo o estudo, concepção e implementação do agente semiótico, que, baseando-se na Engenharia Semiótica, permite a geração de signos, símbolos, conceitos e linguagem, cujos elementos são parte essencial do processo mnemônico de aprendizagem, segundo Vygotsky.

O objetivo deste projeto, como um todo, leva em conta dois pontos importantes:

a) Minsky [MIN 86, Apud in VIC 90a] definiu a IA como:

"a ciência de fazer com que máquinas façam coisas que requereriam inteligência se feitas pelos homens".

b) o objetivo da IA, como comenta Coelho [COE 94], é desenvolver máquinas (programas) que possam resolver problemas que não são acessíveis aos seres humanos.

Dessa forma, estas duas afirmações acima, vêm ao encontro de nosso objetivo neste projeto, que não é substituir o professor, mas sim, utilizar-se das vantagens do uso da EAD, em situações em que, por exemplo, o aluno não pode ir até o professor e/ou vice-versa. Assim, propõe-se um ambiente (*framework*) que possa auxiliar neste processo de ensino-aprendizagem do ser humano a distância, interagindo conjuntamente agentes, alunos e tutores.

Para dar embasamento ao modelo coletivo de EAD, define-se, no capítulo 2, alguns conceitos utilizados na fundamentação teórica, tais como: a seção 2.1 que trata de Educação a Distância; a seção 2.2 apresenta idéias de Sistemas Tutores Inteligentes; a teoria sócio-interacionista inspirada em Vygotsky [VYG 98] e [VYG 98a], e Baquero [BAQ 98], é apresentada na seção 2.3; a seção 2.4 trata de Inteligência Artificial Distribuída; e a seção 2.5, dos conceitos de Engenharia Semiótica.

No capítulo 3, define-se a arquitetura do ambiente educacional envolvendo o modelo pedagógico e o modelo computacional, incluindo uma descrição de todos os agentes envolvidos no projeto.

No capítulo 4, explica-se mais detalhadamente o agente semiótico, apresentando sua arquitetura interna e mostrando a sua ligação com a Engenharia Semiótica. No capítulo 5, detalha-se a implementação com as mensagens passadas entre os agentes, as regras de comportamento, a base de dados do sistema, o ambiente de gerência do material instrucional e o ambiente de educação a distância. Por fim, no capítulo 6, apresenta-se conclusões e trabalhos futuros a serem desenvolvidos.

2 Fundamentação Teórica

2.1 Educação a Distância

O desenvolvimento da computação de alta performance e das tecnologias de comunicação (e-mail, www, *newgroups*, *chat*) tem propiciado melhores ferramentas de "ensino a distância" e não de "aprendizagem a distância". Sem dúvida alguma, o ensino a distância tem trazido vários benefícios, facilitando o acesso à informação, ao estudo individualizado e a coleta de dados. No entanto, seus pressupostos ainda estão focalizando paradigmas clássicos de instrução. O que na verdade ocorre, é que o ensino tem sido interpretado como sinônimo de aprendizagem, quando na verdade não é. O ensino focaliza a transmissão de conhecimentos, disponibilizado pelo professor ou pelos materiais didáticos e bases de conhecimento distribuídas. Mas, o ensino não focaliza o aluno, nem tão pouco o grupo social.

É preciso buscar um modelo social e colaborativo de aprendizagem a distância. Na verdade, admite-se que não há ensino, sem aprendizagem e vice-versa, mas é preciso buscar uma nova abordagem, estabelecer novas estratégias, propiciar o entendimento compartilhado e a solução conjunta de problemas. O papel da tecnologia será essencial a essas mudanças. Mas o que se propõe é utilizá-lo não apenas para informar os indivíduos, mas sobretudo para formá-los de modo permanente e a distância.

Neste sentido, as tecnologias de inteligência artificial distribuída, que serão vistas na seção 2.4, especificamente sistemas multiagentes, além do uso da realidade virtual, dos sistemas multimídia e da Web podem constituir excelentes aliados na construção de um modelo de aprendizagem que favoreça a troca e a interação social entre os indivíduos, propiciando a formação de grupos ou comunidades virtuais de aprendizagem.

2.2 Sistemas Tutores Inteligentes

Num primeiro momento, em termos de evolução da arquitetura dos Sistemas Tutores Inteligentes (STI), pensou-se que os STI seriam modelados como agentes em comunicação com um aluno. Para isso, necessitavam de ter, além de uma interface mais ou menos sofisticada (incluindo uma ou várias modalidades de comunicação), o modelo do seu utilizador (crenças), o conhecimento do domínio a ensinar e as estratégias de ensino (a pedagogia de um professor). Atualmente, segundo Coelho, existe uma outra perspectiva que privilegia o STI como um sistema composto de vários agentes num ambiente de ensino-aprendizagem [COE 94].

Assim, em vez de estar confinado ao isolamento, os STI comportam-se como uma rede de agentes autônomos nesses ambientes. Mas, para manterem um comportamento flexível, exigem uma arquitetura muito mais sofisticada quanto à diversidade dos seus estados mentais e aos poderes do seu raciocínio. Assim, não basta que o STI possua um tipo de raciocínio para manipular ações e conhecimentos. É importante também, que o STI consiga pelo menos rever as suas crenças, quando interatua com os seus alunos, e possuir conhecimentos (dinâmicos e estáticos) sobre si próprio e sobre os outros agentes em presença.

Os Tutores Inteligentes ou Sistemas Inteligentes de Instrução Assistida por Computador (ICAI), segundo Viccari [VIC 90], são sistemas em que a IA desempenha

um papel importante, pois além de permitir uma maior flexibilidade no ensino por computador, possibilita também a participação ativa do aluno e do sistema, gerando um ambiente cooperante para o ensino e a aprendizagem (de ambos os agentes – aluno e sistema).

Assim, Viccari [VIC 90] afirma que um tutor, para ser inteligente, precisa ser flexível e ser capaz de aprender com o meio-ambiente. Deve permitir atualizar constantemente o seu conhecimento a respeito do aluno, recorrendo a operações de particularização ou de generalização, ao comparar o conhecimento apresentado pelos diferentes alunos com o seu próprio conhecimento, além de adaptar sua estratégia de ensino de acordo com o modelo do aluno.

Em termos de flexibilidade de um tutor, entende-se que é o resultado de vários fatores como a capacidade de se reconfigurar automaticamente ao longo de uma sessão de trabalho com o aluno (arquitetura dinâmica); a capacidade para acompanhar (perceber) o raciocínio e a estratégia que determinado aluno está utilizando na resolução de um problema; e a capacidade de se adaptar à linguagem do aluno.

Os STI apresentam uma relação entre os agentes inteligentes (tutor e aluno). Para que esta interação exista, é necessário, segundo Viccari, que o tutor construa o modelo do aluno, transportando o agente humano para dentro da máquina e vice-versa, ou seja, além do modelo do aluno (como o tutor vê, isto é, classifica o conhecimento de seu usuário), também é preciso que o tutor modele como o aluno vê o tutor, isto é, o que o tutor acredita que o aluno acredita sobre o conhecimento e as ações do tutor [VIC 93].

Assim, o modelo do aluno necessita representar as *crenças*, *planos*, *intenções*, *atitudes* e *objetivos*, isto é, as crenças do aluno a respeito do conteúdo que está sendo apresentado; os planos gerados pelo tutor, em função das crenças e dos objetivos (problemas), para ensinar e para acompanhar o raciocínio do aluno (intenção); as intenções representadas pelo plano que o aluno apresenta para solucionar um problema (objetivo); e as atitudes dos agentes refletindo as suas ações e a reflexão que os agentes fazem sobre as suas ações.

Processo este, conhecido como a capacidade de adaptação de um tutor ao seu aluno, onde resultarão duas possibilidades de ação quando existirem diferenças entre os planos do tutor e do aluno:

- o tutor aprende, quando procurará aproximar o seu conhecimento ao conhecimento do aluno; ou
- o tutor ensina, quando procurará aproximar o conhecimento do aluno ao seu conhecimento.

A maneira como o tutor vai aprender, vai depender do tipo de estratégia de aprendizagem a ser utilizada, como, por exemplo, aprendizagem por implantação direta do conhecimento, por instrução, por indução (observação passiva ou experimentação ativa), por analogia, por dedução, ou ainda aprendizagem através de modelos cognitivos. Maiores detalhes sobre este assunto podem ser vistos em Viccari [VIC 90a], Viccari & Oliveira [VIC 92] e Viccari & Giraffa [VIC 96].

Falamos em estratégia de aprendizagem, aquela em que o tutor vai aprender, mas existe também a estratégia de ensino, que, segundo Viccari & Oliveira [VIC 92], é um plano, ou melhor, são esquemas de planos que irão constituir o conhecimento sobre como ensinar, isto é, como gerar uma seqüência de táticas de ensino capazes de apresentar um determinado tópico do domínio ao aluno. Ainda por táticas, entende-se como as ações necessárias para tornar uma determinada estratégia efetiva. Um estudo mais aprofundado sobre estratégias de ensino e táticas pode ser visto em Viccari & Oliveira [VIC 92], Viccari & Giraffa [VIC 96] e Bercht [BER 97].

Segundo Viccari [VIC 90], os tutores inteligentes precisam incentivar a exploração dos conteúdos instrucionais, possuir vários planos de ensino e um modelo para guiar a apresentação do conteúdo instrucional; ser sensível às necessidades do aluno adequando-se às necessidades individuais; dominar, o máximo possível, o assunto que ensina; possuir conhecimento para tentar resolver situações não previstas nas regras existentes e aprender com tais situações; possuir características de ensino assistido (caráter tutorial); possuir mecanismos para a depuração inteligente e a orientação na detecção e eliminação de falhas; permitir a simulação automática e conduzida de problemas; e possuir memória retroativa que descreva o raciocínio (passos) utilizado pelo aluno e pelo tutor durante a exploração de determinado conteúdo instrucional.

Assim, segundo Viccari [VIC 90], os STI são programas que, através da interação com o aluno, modificam suas bases de conhecimento, percebendo as intervenções do aluno e sendo capaz de aprender e adaptar as estratégias de ensino conforme o desenrolar do diálogo com o aluno. Caracterizam-se, principalmente, por construir um Modelo Cognitivo do Aluno através da interação, formulação e comprovação de hipóteses sobre o estilo cognitivo do aluno, seu procedimento, nível de conhecimento do assunto, e ainda baseado nas suas estratégias de aprendizagem e na capacidade de formular estratégias de ensino-aprendizagem adequada ao aluno conforme a necessidade de cada momento.

Por fim, para conseguirmos construir um bom STI com um modelo de aluno ideal, é preciso entender o que acontece na mente do aluno durante a interação no processo de aprendizagem. Assim, é preciso entender o processo e tentar reproduzi-lo o melhor possível dentro da máquina, de acordo com três pontos de vista: a Ciência da Computação (explorando as técnicas de IA), a Psicologia (envolvendo a base teórica da cognição), e o ponto de vista Educacional (pedagógico). A FIGURA 2.1 abaixo, extraída de Viccari [VIC 90], mostra como está integrado os STI em termos de áreas de intervenção e diferenças entre os sistemas CAI e ICAI.

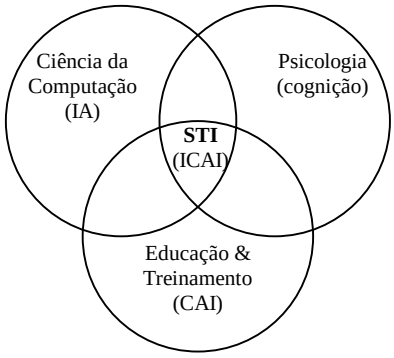

FIGURA 2.1 – Domínio dos STI

Podemos citar alguns trabalhos de STI desenvolvidos pelo grupo de pesquisa (Grupo de Inteligência Artificial - GIA/UFRGS, sob a orientação da Profa. Dra. Rosa Maria Vicari), no qual esta pesquisa também está inserida:

- Viccari [VIC 90] idealizou a construção do Tutor-Prolog que ensina a linguagem Prolog, criando um ambiente de interação cooperativo, ensinando, adaptando-se de acordo com o modelo cognitivo do aluno, cujo modelo do aluno é baseado na revisão de crenças, permitindo também estabelecer uma relação amigável com os seus utilizadores. Adota um modelo de ensino guiado por estratégias, que são utilizadas para a apresentação do material instrucional e para a geração do diagnóstico durante a correção das falhas detectadas.

- Giraffa et al. [GIR 98] e Giraffa [GIR 99] construiram o ambiente MCOE (*Multi-agent Co-operative Environment*), com sua validação apresentada em Giraffa & Vicari [GIR 99a], concebendo um STI composto por um SMA híbrido que possui dois tipos de agente: um reativo (que faz uso de orientação a objeto) e um "kernel" cognitivo (com o uso de estados mentais), onde tem-se um ecossistema formado por peixes, plantas, água e microorganismos, que devem estar em equilíbrio com o auxílio do aluno representado pela mãe natureza, prefeito, pescador e/ou turista, evitando assim a poluição.
- Bica [BIC 99], Bica et al. [BIC 2000] e Hahn et al. [HAN 2000] desenvolveram e avaliaram o Eletrotutor, proporcionando o ensino-aprendizagem de conceitos de Eletrodinâmica e Eletricidade da Física, utilizando STI em ambiente web.
- Silveira [SIL 2000] complementou o estudo de Bica no projeto JADE - Java Agent framework for Distance learning Environments, onde modelou-se agentes aplicados a ambientes inteligentes distribuídos de ensino.
- Moussalle [MOU 96] e Moussalle et al. [MOU 96a] modelaram um STI com agentes utilizando estados mentais, mais especificamente a arquitetura SEM (Sociedade dos Estados Mentais), como pode ser vista na seção 2.4.3, com o objetivo de ensinar algoritmo de divisão de números inteiros através de diálogos exploratórios.
- Móra et al. desenvolveram um ambiente computacional utilizando estados mentais para modelar agentes cognitivos através da linguagem X-BDI. Maiores detalhes podem ser obtidos em Móra et al. [MOR 98] e Móra [MOR 99].

Podemos citar, ainda, outros exemplos de STI que fazem uso de agentes, por exemplo, os agentes pedagógicos animados: Adele, Steve, Vincent, Cosmo, Herman, que podem ser vistos com maiores detalhes no estudo comparativo entre estes agentes feito por Jaques et al. [JAQ 2001].

2.3 Teoria Sócio-Interacionista

Aprendizagem para Vygotsky é resultado da interação social. Mas o que vem a ser interação social? Segundo Clermont & Nelly [CLE 78], podemos dizer que "situações de interações sociais requerem que os sujeitos coordenem entre si as suas ações ou que confrontem os seus pontos de vista, podendo acarretar uma modificação subseqüente da estrutura cognitiva individual".

Um dos conceitos importantes da teoria sócio-interacionista inspirada em Vygotsky é que o funcionamento psicológico está baseado nas relações sociais entre o indivíduo e o meio, em um processo histórico, e que a relação homem-meio é uma relação mediada por sistemas simbólicos, através de instrumentos e signos. Para Vygotsky [VYG 98] e [VYG 98a], e Baquero [BAQ 98], os signos são estímulos artificiais com a finalidade de auxílio mnemônico; eles funcionam como meio de adaptação, dirigido para o controle do próprio indivíduo. O signo é orientado internamente. Já a função de um instrumento é servir como um condutor da influência humana sobre o objeto da atividade; este é orientado externamente. Ambos têm em comum, a função de mediação.

Do ponto de vista conceitual, um outro conceito fundamental em sua teoria é o da Zona de Desenvolvimento Proximal (ZDP). Quando fala-se em ZDP é preciso definir quais são os níveis de desenvolvimento do aluno: Nível de Desenvolvimento Real (NDR) define funções que o aluno já possui; e Nível de Desenvolvimento Potencial (NDP) determina as funções que o aluno pode desenvolver, através da orientação de um

adulto ou a colaboração de colegas mais capazes. A ZDP é a distância entre o nível de desenvolvimento real e o potencial. Além destes conceitos, Vygotsky defende que as funções ocorrem prioritariamente a nível social para depois ocorrerem a nível individual: primeiramente entre pessoas (interpsicológica) e, posteriormente, dentro da pessoa (intrapsicológica). As habilidades psicológicas, da mesma forma, ocorrem inicialmente no relacionamento (intermental) em seguida dentro da criança (intramental).

Este assunto será tratado com mais detalhes na tese de doutorado que está sendo realizada pela Adja Ferreira de Andrade no PGIE/UFRGS.

2.4 Inteligência Artificial Distribuída

A partir da década de 1970, com o surgimento dos Sistemas Tutores Inteligentes, os pesquisadores em Informática na Educação observaram a necessidade de usar técnicas de Inteligência Artificial para tornar os sistemas de ensino mais flexíveis e adaptados aos seus usuários. Atualmente, o enfoque colaborativo dado à educação a distância, visto na seção 2.1, desencadeou um grande número de problemas e atividades em que a tecnologia de agentes pode ser bem empregada, tanto auxiliando e monitorando os alunos como também provendo informações ao professor.

As questões que antes tratavam de estratégias de resolução de problemas em um ambiente de processamento distribuído, passaram então a ser vistos por meio de sistemas compostos de múltiplos subsistemas com capacidade de resolver problemas diversos de forma cooperativa.

Isto vem ao encontro com a seguinte afirmação de Minsky [MIN 86, Apud in COE 94]:

“Chamarei Sociedade da Mente a um esquema no qual cada mente é feita com muitos processos pequeninos, chamados agentes. Cada agente só pode fazer coisas simples que não exijam qualquer mente ou pensamento. No entanto, quando juntamos estes agentes em sociedades – em modos especiais – isto conduz à verdadeira inteligência”.

Assim, na Inteligência Artificial Distribuída (IAD), a inteligência é vista como emergente da ação e interação de agentes inteligentes autônomos.

Podemos dizer, ainda, que as ações e resoluções de problemas não envolvem apenas uma única entidade ou pessoa, mas sim, pressupõem a interação entre os mesmos. Isto vem ao encontro com a visão de Vygotsky, conforme explicado na seção 2.3, onde o processo de aquisição de conhecimentos e capacidades inteligentes individuais forma-se em conjunto com o fenômeno relacionado com a interação social.

A IAD tornou-se, nos últimos anos, um domínio de pesquisa muito promissor. Enquanto estudos clássicos de Inteligência Artificial (IA) tomam como modelo de inteligência o comportamento individual humano, cuja ênfase é colocada em representação de conhecimento e métodos de inferência, o modelo de inteligência utilizado em IAD é baseado no comportamento social, sendo a ênfase colocada em ações e interações entre agentes, segundo Sichman [SIC 92]. Em IAD, usa-se a expressão agente no lugar de sistemas baseados em conhecimento ou de outra expressão para descrever entes que participam ativamente na solução de um problema.

Podemos dizer, assim, que a IAD, é a área da IA que trata das sociedades de agentes artificiais, cuja sociedade se propõe a solucionar, de maneira cooperativa e distribuída, problemas que não poderiam ser resolvidos de forma eficiente por um

sistema computacional centralizado. Nela, os agentes interagem entre si encontrando a melhor solução para um problema complexo.

Sichman et al. [SIC 92] dividem a IAD em dois campos principais, que são:

- Resolução de Problemas Distribuídos - DPS (*Distributed Problem Solving*) - que trata da existência de algum problema para resolver ou uma tarefa a ser executada, e o desenvolvedor projeta um sistema composto de múltiplos agentes para resolver o problema. Segundo Jennings et al. [JEN 98], considera como um problema particular pode ser resolvido por um número de módulos (nodos), que cooperam na divisão e compartilhamento do conhecimento sobre o problema e sua solução envolvida. Num sistema DPS puro, todas estratégias de interação são incorporadas como uma parte integral do sistema.
- Sistemas Multiagentes (SMA) - existe uma sociedade de agentes autônomos que se organizam para resolver o problema solicitado, cuja sociedade existe independente de qualquer problema ou tarefa, ou seja, segundo Jennings et al. [JEN 98], trata de uma coleção de agentes autônomos preexistentes que visam na solução de um dado problema, não necessariamente um só (podendo ser individuais ou coletivos).

Características de DPS

Pontos importantes da abordagem DPS segundo Moussalle [MOU 96]:

- os agentes não precisam representar suas habilidades, pois elas são representadas implicitamente pelo projetista;
- o projetista, que não é um agente, descreve e decompõe (ou distribui) a tarefa para os agentes;
- não surge conflito entre os agentes se a tarefa for bem dividida;
- o sistema é fechado, não permitindo que novos agentes sejam dinamicamente adicionados à sociedade.

Ainda em Alvares & Sichman [ALV 97] cita-se:

- o problema é resolvido por um conjunto de agentes, fisicamente distribuídos em diversas máquinas conectadas via rede, cujos agentes são concebidos para solucionar um determinado problema particular;
- uma organização é concebida para restringir o comportamento destes agentes. Esta organização, na maioria das vezes, é completamente definida durante a fase de concepção do sistema;
- a interação entre os agentes é realizada ou por troca de mensagens, ou por meio do compartilhamento de dados. A estrutura destas trocas é quase sempre definida completamente durante a fase de concepção do sistema, sendo intimamente relacionada ao modelo subjacente (como por exemplo, o quadro-negro) e ao problema que o sistema deve resolver;
- os agentes são executados de modo concorrente, aumentando assim a velocidade de resolução;
- os agentes cooperam, dividindo entre si as diversas partes do problema original (subproblemas, tarefas), podendo, inclusive, aplicar diferentes estratégias de resolução para uma mesma tarefa;

- existe a noção de um controle global, na maior parte dos casos, implícito ao agente, garantindo, assim, um comportamento global coerente do sistema conforme a organização inicialmente prevista.

2.4.1 Sistemas Multiagentes

O objetivo não é abordar tudo que existe sobre SMA, e sim dar uma idéia sobre alguns pontos considerados importantes para a compreensão deste trabalho. Maiores detalhes sobre SMA, onde existem muitas outras referências, podem ser consultados, por exemplo, em Wooldridge & Jennings [WOO 95], Jennings et al. [JEN 98] e Sycara [SYC 98].

Quando um sistema é formado por mais de um agente, ele é conhecido como SMA. Os agentes são entidades autônomas que possuem conhecimento meta-nível de si próprio e dos outros agentes na sociedade e, portanto, colaboram uns com os outros para atingirem um objetivo comum em um mesmo ambiente.

Em um SMA, os agentes devem possuir algumas capacidades específicas para interagirem num mesmo ambiente. Além disso, eles devem ser capazes de se comunicar possuindo, para tanto, uma linguagem de alto nível. Cada agente deverá possuir conhecimento e habilidades para executar uma determinada tarefa, podendo cooperar ou não, para atingir um objetivo global.

A tecnologia de agentes pode trazer significativo suporte à aprendizagem na Web. As características de autonomia, colaboração e aprendizagem podem auxiliar na construção de um modelo de aprendiz e auxiliar na interação entre os alunos, incentivando o seu desenvolvimento sócio-cognitivo.

Demazeu & Müller [DEM 89] e Sichman et al. [SIC 92] colocam algumas considerações importantes para a abordagem de SMA:

- os agentes devem ser capazes de decompor as tarefas baseados no conhecimento que eles possuem de si próprio e dos outros agentes;
- como os agentes são autônomos, eles podem possuir metas próprias e decidirem o que fazer a qualquer momento, podendo cooperar ou não com os outros para atingir o objetivo comum;
- os agentes possuem capacidade para resolver seus problemas e os problemas que surgirem no ambiente;
- o sistema é aberto, permitindo que agentes entrem e saiam da sociedade a qualquer momento. Assim, os agentes devem ser capazes de modificar o conhecimento que possuem dos outros agentes do ambiente;
- os agentes devem ser capazes de reconhecer modificações no ambiente quando estas ocorrerem, alterando sua representação interna do ambiente.

Além destas considerações, complementa-se, segundo Moussalle [MOU 96]:

- a distribuição de ações para a realização da tarefa é feita pelos agentes e não pelo projetista;
- surgem vários conflitos devido à existência de objetivos locais e globais, devendo existir, assim, uma conversa complexa entre os agentes, para que possa determinar o papel de cada um, permitindo êxito na execução de uma dada tarefa.

Possui ainda as seguintes características, segundo Alvares & Sichman [ALV 97]:

- os agentes são concebidos independentemente de um problema particular a ser resolvido. O projeto de um agente deve resultar numa entidade capaz de

realizar um determinado processamento, e não numa entidade capaz de realizar este processamento exclusivamente no contexto de uma aplicação alvo particular;

- a concepção das interações também é realizada independentemente de uma aplicação-alvo particular. Busca-se desenvolver protocolos de interação genéricos, que possam ser reutilizados em várias aplicações similares. Um exemplo de um tal protocolo seria, por exemplo, um protocolo de apresentação de um agente quando este ingressa numa sociedade. Obviamente, um protocolo deverá ser instanciado com dados do domínio do problema para poder ser efetivamente utilizado numa aplicação;
- a mesma filosofia anterior pode ser estendida ao projeto das organizações. Normalmente, se distingue as funcionalidades necessárias a uma resolução particular dos agentes que irão efetivamente implementar tais organizações;
- durante a fase de resolução, os agentes utilizam suas representações locais dos protocolos de interação e das organizações para raciocinar e agir. Deste modo, não existe um controle global do sistema, este é implementado de forma totalmente descentralizada nos agentes.

Sycara [SYC 98] e Jennings et al. [JEN 98] também apresentam algumas características dos SMA:

- cada agente tem capacidade ou informação incompleta para resolver um problema e, assim, possui um ponto de vista limitado;
- não existe um controle global no sistema;
- os dados estão descentralizados;
- a computação é assíncrona.

Segundo Jennings et al. [JEN 98], sistemas multiagentes são ideais para representar problemas que têm múltiplos métodos de resolver problemas, múltiplas perspectivas e/ou múltiplas entidades para resolver problemas. Tais sistemas têm as vantagens dos sistemas tradicionais ou distribuídos e solução de problemas concorrentes, mas tem a vantagem adicional de sofisticados padrões de interações. Exemplos de tipos comuns de interações incluem:

- cooperação (trabalhando junto em direção a um objetivo comum);
- coordenação (problema de organização que resolve atividade de forma que interações prejudiciais são evitadas ou são exploradas interações benéficas);
- negociação (chegada de um acordo que é aceitável a todas as partes envolvidas).

É esta flexibilidade e natureza de alto-nível destas interações que distingue sistemas multiagentes de outras formas de software e que provê o poder implícito do paradigma.

SMA pode ser definido como uma rede unida livremente de resolvedores de problema que trabalham juntos para resolver problemas que estão além das capacidades individuais do conhecimento de cada resolvedor de problema. Estes resolvedores de problemas - agentes - são autônomos e podem ser heterogêneos em sua natureza.

2.4.2 Agente

Embora o termo “agente” tenha sido usado com freqüência na IA, sendo uma noção central e fundamental para área de IAD, não há um consenso uniforme e preciso para o que um agente é ou significa. Cada grupo de pesquisa segue uma determinada linha apresentando sua definição personalizada do termo agente de acordo com seus

próprios objetivos, mas todos de uma maneira geral definem como uma entidade capaz de executar uma tarefa ou um conjunto de tarefas:

- Giraffa [GIR 99] defende que este termo vem sendo utilizado para denotar desde simples processos de hardware e/ou software até entidades sofisticadas com capacidade de realizar tarefas complexas. Mais especificamente, "agente" pode ser entendido como uma entidade que exibe alguns aspectos da inteligência humana.
- Russel & Norvig [RUS 95] definem um agente como um sistema com capacidade de percepção através de sensores, e que agem em um dado ambiente através de atuadores.
- Jennings et al. [JEN 98] afirmam que um agente é um sistema de computador, situado em algum ambiente, que é capaz de ação autônoma flexível para atingir os objetivos para os quais foi projetado.
- Segundo Oliveira [OLI 95], um agente pode ser definido como uma entidade capaz de perceber e executar ações que causam trocas no ambiente e/ou nos estados internos dos agentes com a finalidade de atingir determinadas metas. A existência de metas independentes é uma característica natural e desejável em sociedades de agentes autônomos, cuja existência é independente de qualquer problema ou tarefa.
- De um modo geral, para a IA, o termo agente é mais freqüentemente associado a uma entidade que funciona contínua e autonomamente em um ambiente no qual existem outros processos, objetos e agentes, segundo Corrêa [COR 94].
- O agente é uma entidade autônoma que possui conhecimento meta-nível de si próprio e dos outros agentes na sociedade e, portanto, colabora com os outros para atingir um objetivo comum em um mesmo ambiente, segundo Jaques & Oliveira [JAQ 98].
- Segundo Coelho, um agente é uma entidade real ou virtual, situada num certo ambiente, onde pode realizar ações, sendo capaz de percepcionar e representar parcialmente este ambiente, capaz de comunicar com os outros agentes e que possui um comportamento autônomo como conseqüência das suas observações, conhecimento e interações com os outros agentes [COE 94].
- O agente também pode ser uma entidade à qual se atribuem estados, denominados de estados mentais, tais como crenças, decisões, capacidades, objetivos e compromissos (conceitos análogos ou similares aos humanos), segundo Shoham [SHO 93].

Wooldridge & Jennings [WOO 95] estabelecem quais são as propriedades que um sistema deve apresentar para que possa ser considerado um agente:

- **Autonomia** – agentes operam sem a intervenção direta de humanos ou outros agentes, e possuem a capacidade de controlar suas próprias ações e estados internos;
- **Habilidade social** – agentes interagem com outros agentes artificiais e humanos através de algum tipo de linguagem de comunicação entre agentes, que, segundo Jennings et al. [JEN 98], têm a finalidade de resolver seu próprio problema, ajudar outros com suas atividades ou ainda ajudar os outros reciprocamente;

- **Reatividade** – os agentes percebem seu próprio ambiente ou os outros agentes e respondem de forma oportuna a mudanças que ocorrem ao seu redor;
- **Pró-atividade** – agentes não agem simplesmente em resposta ao seu ambiente, mas possuem a capacidade de exibir comportamentos dirigidos por objetivos, sendo capaz de tomar iniciativa apropriada.

Além destas, Giraffa [GIR 99] acrescenta, ainda, as seguintes propriedades:

- **Contínuo** - capacidade de permanecer no ambiente, através de processos continuamente rodando;
- **Comunicativo** – capacidade de trocar informações com os outros agentes;
- **Aprendizagem** – capacidade de aprender com as informações oriundas do ambiente ou de outros agentes;
- **Mobilidade** – capacidade de se deslocar para ambientes diferentes do original;
- **Flexibilidade** – capacidade de aceitar a intervenção de outros agentes;
- **Racionalidade** – vai depender do grau de sucesso e percepção que o agente possui;
- **Adaptabilidade** – habilidade de se adaptar a modificações no ambiente, alterando suas ações (planos previamente concebidos) ou aprendendo através da interação com o ambiente.

Quem determina quais as propriedades que o agente deve ter é o ambiente e seu respectivo projeto, e para certos tipos de aplicações, alguns atributos serão mais importantes que outros. Porém, acredita-se que é a presença de todos os atributos em uma única entidade de software que provê o poder do paradigma de agente, e o qual distingue sistemas de agentes, de paradigmas de software relacionados - como sistemas orientados a objetos, sistemas distribuídos, e sistemas especialistas.

A comunicação entre os agentes segue um protocolo e pode se dar através da simples troca de mensagens ou ter uma forma mais elaborada (diálogos). A negociação, a cooperação e a solução dos conflitos são fenômenos centrais nas atividades que ocorrem através de diálogos.

Conforme a definição de Shoham [SHO 93], já mostrada anteriormente, agente é uma entidade à qual se atribuem estados, cujo estado é visto como consistindo de componentes mentais como: crenças, capacidade, decisão (escolha) e obrigação (compromisso), modelados de forma análoga aos similares estados mentais humanos. Por essa razão, o estado de um agente é chamado de estado mental. Estes componentes devem ser definidos numa forma precisa e possuir uma correspondência direta com o seu uso no senso comum.

Podemos chamar este enfoque de **enfoque mentalístico**, pois nesta definição de agente, o que faz qualquer sistema um agente é precisamente o fato deste sistema poder ser analisado e controlado em termos de estados mentais.

Assim, a questão do que realmente é um agente, é trocada pela questão de quais entidades podem ser vistas como possuindo estados mentais. Isto leva à distinção entre a legitimidade de atribuir qualidades mentais para as máquinas e a sua utilidade. Atribuir estados mentais para a máquina é legítimo sempre que existir uma correspondência entre eles (estados mentais) e seus correspondentes no senso comum; e é útil, quando ajuda a compreender e controlar a estrutura do sistema, o seu comportamento, o seu passado, o seu presente e o seu futuro, e ainda reparando-a ou melhorando-a.

Agente autônomo é, segundo Sichman et al. [SIC 92], o agente que tem sua própria existência independente da existência prioritária de qualquer problema a ser resolvido. Os agentes coexistem num ambiente comum e cada um pode colaborar com os outros para atingir seus objetivos.

O agente autônomo tem a capacidade de atuar inteligente e racionalmente, isto é, tem o controle das suas ações e estados internos. Sua autonomia pode variar dependendo do tipo de arquitetura em que foi construído.

Agente inteligente é aquele que se comporta de forma flexível em função do ambiente, segundo Moussalle [MOU 96], exibindo um comportamento adaptativo e operando em tempo real num ambiente rico e complexo. Todo o agente que pertence à classificação citada acima utiliza enorme quantidade de conhecimento, usando uma linguagem natural e/ou artificial e aprende a partir do ambiente, adquirindo novas capacidades. Tal tipo de agente vive autonomamente dentro de uma comunidade artificial sempre atento sobre si e sobre o que o rodeia.

Agentes pedagógicos são, segundo Giraffa [GIR 99], aqueles utilizados em sistemas que utilizam o paradigma de agentes desenvolvidos para fins educacionais, podendo atuar como tutores virtuais, estudantes virtuais, ou ainda companheiros virtuais de aprendizagem, tendo como objetivo auxiliar os estudantes no processo de ensino-aprendizagem.

Os agentes pedagógicos dividem-se em agentes *goal-driven* (guiados por objetivos) que são os tutores, mentores e assistentes; e agentes *utility-driven* (guiados pela sua utilidade no ambiente) que são os agentes que realizam tarefas auxiliares ligadas às atividades pedagógicas, executando tarefas para o estudante ou para o professor como por exemplo agendamento de encontro de grupos, lembrete de atividades a serem entregues (temas, exercícios), podendo atuar, tanto em ambientes de rede, como em ambientes Web [GIR 99].

Agente cognitivo é um agente deliberativo, que possui objetivos explícitos e tomam decisões levando-os em consideração, possui uma grande granularidade, é inteligente e está baseado no modelo de organização social das sociedades humanas. Pode raciocinar sobre as ações do passado e fazer planos para futuras ações. Seu enfoque principal é a representação explícita do ambiente e dos poucos membros da sociedade.

Segundo Corrêa [COR 94], os agentes cognitivos são agentes que possuem a capacidade de compreender o ambiente onde estão atuando, discriminar e identificar objetos e indivíduos, propriedades e relações que ocorrem entre os indivíduos num determinado ambiente e possuem meios para manipular, extrair e utilizar a informação obtida.

Segundo Jaques & Oliveira [JAQ 98], esses agentes também podem ser ditos sociais porque, além de manipular o seu conhecimento, eles conhecem as crenças, objetivos e motivações dos agentes que o cercam. Para formar um grupo social unindo um conjunto de agentes cognitivos, é necessário que algumas questões sejam tratadas, tais como: o tipo de organização adotada, a cooperação entre os agentes, a negociação entre os agentes para divisão de tarefas e como os agentes se comunicarão.

As principais características dos agentes cognitivos, segundo Alvares & Sichman [ALV 97] são:

- mantêm uma representação explícita de seu ambiente e dos outros agentes da sociedade;
- podem manter um histórico das interações e ações passadas, isto é, têm memória do passado;

- a comunicação entre os agentes é feita de modo direto, através do envio e recebimento de mensagens;
- seu mecanismo de controle é deliberativo, ou seja, tais agentes raciocinam e decidem sobre quais objetivos devem alcançar, que planos seguir e quais ações devem ser executadas num determinado momento;
- seu modelo de organização é baseado em modelos sociológicos, como as organizações humanas;
- uma sociedade contém tipicamente poucos agentes, na ordem de dezenas.

Agente reativo é um tipo de agente simples, não inteligente, que não tem memória, isto é, não guarda as ações ocorridas no passado e não prevê as ações possíveis para o futuro. Não possui um modelo de comunicação de alto nível, nem uma representação explícita do ambiente e muito menos um grande número de membros da sociedade. Possui a capacidade de fazer trocas quando o ambiente muda e compreende o comportamento dos outros membros quando existem trocas, pois seu enfoque principal está no comportamento. São agentes não deliberativos (seus objetivos não são explícitos) seu comportamento é emergente e possuem uma fina granularidade.

Principais características dos agentes reativos, segundo Alvares & Sichman [ALV 97]:

- não há representação explícita de conhecimento - o conhecimento dos agentes é implícito e se manifesta através do seu comportamento;
- não há representação do ambiente - o seu comportamento se baseia no que é percebido a cada instante do ambiente, mas sem uma representação explícita deste;
- não há memória das ações - os agentes reativos não mantêm um histórico de suas ações, de forma que o resultado de uma ação passada não exerce nenhuma influência direta sobre suas ações futuras;
- organização etimológica - a forma de organização dos agentes reativos é similar a dos animais, em oposição à organização social baseada em sociedades humanas dos sistemas cognitivos;
- grande número de membros – têm, em geral, um grande número de agentes, da ordem de dezenas, centenas ou mesmo milhares de agentes;
- possuem comportamento simples do tipo estímulo-resposta.

Agentes globais e locais: Corrêa, em seu trabalho [COR 94], classificou os agentes autônomos em agentes globais e locais para facilitar o tratamento entre eles e observar melhor as interações e a troca de mensagens que ocorriam. Estes dois tipos de agentes conservam as mesmas características dos agentes cognitivos autônomos. O agente global é o agente que emerge da sociedade formada pelos agentes locais e é visto como um todo, com todas as propriedades peculiares a um agente. Cada agente global é composto por estados mentais (crenças, desejos, intenções, expectativa).

2.4.3 Arquiteturas de Agentes

O que caracteriza um agente, são as interações que este realiza com o mundo[1] e, também, quais e como são os processos internos que possibilitam a realização destas interações. Entendemos arquitetura do agente como sendo a especificação de quais e como são estes processos internos.

[1] *mundo* pode ser definido como sendo a descrição completa e instantânea do ambiente onde se encontra o agente, segundo Viccari & Giraffa [VIC 96].

Entende-se por interações, as ações ou seqüência de ações físicas que os agentes autônomos realizam no mundo ou os diálogos que eles mantêm entre si.

Dentre os vários tipos de arquiteturas de agentes, em enfoques os mais variados, destacamos, conforme apresentado em Corrêa [COR 94], Coelho [COE 94] e Viccari & Giraffa [VIC 96]: arquiteturas deliberativas, não deliberativas e híbridas; além da arquitetura SEM apresentada por Corrêa [COR 94] e utilizada em Viccari & Giraffa [VIC 96], Moussalle [MOU 96] e Giraffa [GIR 99].

Arquiteturas deliberativas

São aquelas nas quais a escolha da ação é feita através de uma deliberação explícita sobre diferentes opções, por exemplo, considerando algum plano ou ainda uma função que avalia uma ação conforme sua utilidade; o agente conhece o que está fazendo no momento de realizar a ação, mas, se não tiver metaconhecimento, não sabe como reagir diante de uma situação inesperada.

Nestas arquiteturas, as interações possuem uma grande flexibilidade, facilitando a comunicação entre os agentes e possibilitando aprendizagem e raciocínio que são capacidade de alto nível dos agentes.

Estas arquiteturas também podem ser chamadas, na literatura, de cognitivas.

Arquiteturas não deliberativas

São aquelas em que a escolha da ação está situada na ocorrência de um conjunto de eventos que acontecem no ambiente do agente; este não pode descobrir alternativas para o seu comportamento diante de uma situação diferente dos seus objetivos iniciais. Em geral, elas são construídas a partir de mecanismos de controle simples, tais como, por exemplo, máquinas de estados finitos ou conjunto de regras do tipo estímulo-resposta dependentes de um domínio, e inspiram-se na hipótese de Simon citado por Coelho [COE 94], que diz: “o comportamento complexo de um agente não tem necessariamente de ser um produto de um projeto interno complexo, isto é, a complexidade do seu comportamento é um reflexo da complexidade do ambiente onde ele opera”.

Todas as decisões de controle necessárias são tomadas em tempo real, isto é, a ação do agente ocorre no instante em que ele percebe algum sinal ou estímulo do meio ambiente, agindo num curto espaço de tempo, e baseado numa quantidade de informação restrita e pequena em relação à situação imediata, usualmente, a informação que é correntemente disponível pelos seus sensores.

Na literatura, estas arquiteturas são referidas também pelo nome de “reativas”, “baseadas em comportamento” ou ainda “tropistas”[2].

Arquiteturas híbridas

Um problema central que ocorre com agentes cuja arquitetura é deliberativa, é, tipicamente, a sua incapacidade de agir apropriadamente quando surge uma situação imprevista em relação à qual ele tem que tomar uma rápida decisão (por exemplo, o tempo demasiado de análise de situações ou estados, da construção ou reutilização de

[2] O termo tropista é derivado de “tropismo” que é a tendência que um animal ou planta tem de agir em resposta à um estímulo externo, segundo Corrêa [COR 94].

planos, frente a mudanças que ocorrem no meio ambiente). Por outro lado, um problema típico dos agentes, cuja arquitetura é não deliberativa, é ser incapaz de descobrir alternativas para o seu comportamento quando a situação do mundo diverge bastante de seus objetivos iniciais (objetivos pré-programados).

As arquiteturas híbridas foram projetadas para superar estes problemas. São aquelas em que a escolha da ação é feita usando alguma combinação entre as técnicas usadas nos tipos deliberativos e não-deliberativos.

O objetivo destas arquiteturas é dotar o agente com capacidades reativas apropriadas e aperfeiçoar a capacidade de raciocínio e planejamento do agente com um tempo de resposta aceitável (em relação às arquiteturas deliberativas). São capazes de reagir a situações inesperadas, pois possuem mecanismos para produzir reações a diferentes situações e construir planos usando os recursos computacionais disponíveis.

Arquitetura BDI

Um segmento da pesquisa em IA tem explorado modelos de agentes baseados em crenças, desejos e intenções. As arquiteturas que seguem este paradigma são conhecidas como arquiteturas BDI (*Belief*, *Desire*, and *Intention*). As idéias básicas da abordagem BDI são descrever o processamento interno do estado de um agente utilizando um conjunto de estados mentais (crenças, desejos e intenções) e definir uma arquitetura de controle através da qual o agente seleciona racionalmente o curso de suas ações. Algumas abordagens de arquitetura BDI agregam as noções de planos e objetivos.

Estas arquiteturas são modeladas ou inspiradas no raciocínio prático dos seres humanos. Tipicamente, teorias do raciocínio prático fazem uso de psicologia do povo, por meio da qual a ação é entendida pela atribuição de atitudes como crenças, desejos, intenções, etc. Ações humanas podem ser pensadas como fruto das interações destas atitudes mentais.

Intuitivamente:

- crenças correspondem à informação que o agente tem sobre seu ambiente;
- desejos representam opções possíveis para o agente - estados possíveis diferentes de ocupação (negócio) que o agente pode escolher para comprometer;
- intenções representam estados de ocupação que o agente escolheu e comprometeu recursos para execução.

Um raciocínio prático do agente envolve repetidamente a atualização de crenças das informações do ambiente, decidindo que opções estão disponíveis, filtrando estas opções para determinar novas intenções e agindo na base destas intenções. Entre os sistemas de agentes BDI que tem sido implementados, o mais conhecido é, provavelmente, PRS - *Procedural Reasoning System* (sistemas de raciocínio procedural), desenvolvido a partir de lógicas BDI, segundo Jennings et al. [JEN 98].

Maiores detalhes e um formalismo lógico utilizado na modelagem de agentes com arquitetura BDI podem ser vistos em Rao & Georgeff [RAO 91]. Podemos ver também um exemplo de utilização de arquitetura BDI em um sistema de controle de tráfego aéreo desenvolvido por Rao & Georgeff [RAO 95].

Arquitetura SEM

A idéia principal desta abordagem mentalística, segundo Corrêa [COR 94], se concentra no fato de que o agente cognitivo possui estados internos que se relacionam

com o estado do ambiente com o qual interage. Estes estados seriam correspondentes aos estados humanos, que apresentam um vínculo com o mundo em termos da sua existência e significância.

A arquitetura SEM, Sociedade dos Estados Mentais, desenvolvida por Corrêa e apresentada em [COR 94], é uma arquitetura genérica de agente cognitivo, deliberativo e autônomo, cuja característica fundamental é que a especificação do agente é feita somente através dos estados mentais crença, desejo, intenção e expectativa. As expectativas não fazem parte da arquitetura BDI originalmente conforme apresentado por Bratman [BRA 89] e Shoham [SHO 90], mas foram incorporadas ao conjunto de estados mentais abordados em Corrêa [COR 94], sendo tratadas como crenças futuras. Isto traz uma vantagem sobre as arquiteturas BDI, pois o fato de ter-se o estado mental *expectativa*, permite uma maior flexibilidade, perfeição e comportamentos mais complexos, segundo Corrêa et al. [COR 98].

Toda a teoria sobre o comportamento do agente fundamenta-se no fato do projetista necessitar apenas especificar quais são os estados mentais do agente. Assim, a compreensão de um agente é obtida através da determinação dos seus estados mentais, e a estrutura das interações dos agentes é construída relacionando as noções chaves de arquitetura, estados mentais e contexto.

Na arquitetura SEM, o comportamento do agente que é representado pelas ações através das quais ele interage com outro agente e no mundo, resulta das relações causais entre estes estados mentais que são o núcleo a partir do qual é definida a arquitetura do agente. Assim, em Corrêa [COR 94], a arquitetura de agente (chamada de global) é composta por quatro agentes locais, cada um deles correspondendo, respectivamente, aos estados mentais: crença, desejo, intenção e expectativa. Um resultado fundamental, é que as interações do agente global emergem das interações destes agentes locais que comunicam-se através de mensagens.

Embora restringindo a estes quatro estados mentais, este conjunto pode ser ampliado, contribuindo para uma maior riqueza de possibilidades das interações. Por exemplo, Shoham [SLO 90] sugere os estados mentais *motivação* e *emoção*; ou, ainda, Corrêa & Coelho em [COR 98a] e [COR 98b] sugerem outros estados mentais como a *esperança* e a *necessidade*, além de sugerirem também alguns atributos que definem características próprias dos estados mentais como insatisfação, incerteza, urgência, intensidade, importância e insistência. Podemos citar, ainda, a utilização de alguns estados mentais afetivos como esforço, confiança e independência que foram sugeridos e utilizados nos trabalhos de Bercht et al. [BER 99] e Vicari [VIC 2000].

Muitos trabalhos foram feitos utilizando-se a construção de agentes com o uso de estados mentais como podem ser melhor visualizados em Móra et al. [MOR 98], Móra [MOR 99], Giraffa et al. [GIR 98], Giraffa & Viccari [GIR 98a], Giraffa [GIR 99], Moussalle [MOU 96], Moussalle et al. [MOU 96a], Vicari [VIC 2000] entre outros.

Os estados mentais desejo, intenção e expectativas pertencem a uma classe mais ampla de estados chamada de *motivadores*, os quais conforme Sloman [SLO 90] são mecanismos e representações que tendem a produzir, modificar ou selecionar ações, à luz das crenças.

Crenças representam as informações que o agente tem sobre o ambiente e sobre si próprio. Assume-se que o agente atualiza continuamente suas crenças para refletir mudanças que detecta no ambiente e que sempre quando uma nova crença é somada ao conjunto de crenças, a consistência é mantida.

Na visão de Bratman [BRA 89], crenças representam a interpretação (visão) que o agente possui sobre o ambiente em que ele se encontra.

Para Corrêa [COR 94], crença é um estado mental intencional fundamental para as interações dos agentes, com noção idêntica a de conhecimento, cujo conteúdo é uma proposição.

Desejos são relacionados eventualmente ao estado de mundos que o agente quer provocar, ou seja, representam uma situação ou um conjunto de situações que o agente gostaria que estivessem presentes no mundo. Desejos não dirigem necessariamente o agente para agir. Isto é, o fato de um agente ter um desejo não significa agir para o satisfazer. Significa que antes de um determinado agente decidir o que fazer, ele passa por um processo de racionalização e confronta os seus desejos (o estado de eventos que quer provocar) com suas convicções (as circunstâncias atuais e restrições que o mundo impõe). O agente escolherá os desejos que são possíveis de acordo com algum critério. Por exemplo, o agente escolhe os desejos em função das crenças válidas no momento a respeito do ambiente.

O desejo é um estado mental intencional, motivador e assumimos que possui as seguintes características:

- representa uma situação ou um conjunto de situações que o agente gostaria em que o mundo estivesse;
- pode estar em conflito com as crenças do agente;
- podem existir simultaneamente desejos conflitantes;
- não causam diretamente as ações, mas são potenciais para as suas ocorrências.

As relações causais podem ocorrer entre as situações de percepção, crença e desejo de um agente formando uma intrincada teia de interdependências, fazendo surgir ou desaparecer crenças e desejos, ou alterando os valores atribuídos a estes estados mentais.

Intenções são caracterizadas por uma escolha de um estado de eventos a alcançar. Assim como os desejos, as intenções contêm a representação dos estados que o agente quer que se verifiquem.

Segundo Bratman [BRA 89], desde que agentes são assumidos a serem recursos limitados, eles não podem avaliar continuamente suas crenças e desejos concorrendo para agir racionalmente. Depois de algum raciocínio, agentes têm que comprometerem-se para um conjunto de escolhas. Esta escolha seguida por um compromisso que caracterizam as intenções.

Assim, as intenções são vistas como um compromisso que o agente assume com um específico futuro possível. Isso significa que, diferentemente dos desejos, uma intenção não pode ser contraditória com outras intenções. Além disso, significa que não seria racional para um agente agir para alcançar estados incompatíveis. Intenções deveriam ser apoiadas pelas convicções do agente, isto é, não seria racional para um agente pretender algo que não acredita ser possível. Uma vez que uma intenção é adotada, o agente procurará satisfazer aquela intenção e planejará ações para realizá-la. O replanejamento de ações ocorre sempre quando acontece um fracasso. Essas ações também devem ser adotadas como intenções por parte dos agentes.

A definição de intenções obriga criarmos restrições para a racionalidade: um agente não deveria pretender algo num tempo passado; um agente não deveria pretender algo que acredita já estar satisfeito ou que será satisfeito sem esforços por parte do agente; e um agente só pretende algo que acredita ser possível de ser alcançado. Nesse último caso, pode haver um curso de ações que conduzem ao estado intencional associado ao evento. Quando projetamos um agente, nós especificamos as suas crenças e os seus desejos. O agente é que vai escolher apropriadamente suas intenções a partir

de seus desejos. Essas restrições de racionalidade também devem ser garantidas durante esse processo de seleção, segundo Móra et al. [MOR 98].

As intenções têm as seguintes propriedades:

- se um agente tem a intenção de **P** então acredita que **P** é possível. Isto é, ele possui uma estratégia para alcançar **P** ou acredita que é possível ter uma estratégia para alcançar **P**;
- as intenções são compatíveis, isto é, um agente não pode ter, num mesmo momento, a intenção de **P** e **P'** ou acreditar que vai realizar uma interação tal que **P** e **P'** aconteçam no curso de eventos, sendo **P** e **P'** proposições incompatíveis;
- uma intenção somente pode ser causada por um desejo;
- uma intenção pode causar outra intenção.

Bratman [BRA 89] apresenta a idéia de intenção como um estado mental associado a planos de coordenação e a ação intencional. Intenções direcionadas para o futuro são elementos típicos de grandes planos e auxilia a coordenação das atividades ao longo do tempo. Para coordenar as atividades do agente, o plano deve ser consistente internamente, isto é, deve ser possível realizá-lo por completo.

Expectativa é o quarto estado mental associado aos agentes cognitivos que é tratado como uma crença futura do agente.

Segundo Corrêa [COR 94], o princípio de relevância condicional (dado o primeiro, o segundo é esperado) pode ser estendido a qualquer seqüência de ações realizadas pelo agente: a realização de uma ação pode causar um conjunto de expectativas a respeito das possíveis situações que vão ocorrer no futuro.

A expectativa apresenta as propriedades descritas a seguir:

- uma crença e a realização de uma ação podem causar uma expectativa;
- uma expectativa pode causar uma ação;
- uma expectativa pode causar uma crença;
- uma expectativa pode causar um desejo.

As expectativas têm a função de contribuir para aumentar a eficiência da satisfação das intenções visto que a reação do agente para satisfazer uma expectativa quando ela é frustrada, evita a necessidade imediata de rever a estratégia traçada inicialmente, mesmo que esta estratégia seja parcial.

Uma outra função das expectativas, no sentido de otimizar a satisfação das intenções, é restringir o espaço de possibilidades para as estratégias que serão escolhidas conforme as situações esperadas. Nas interações entre os agentes, por exemplo, as expectativas funcionam como normas para os seus comportamentos. Um agente **A** vai ter expectativas a respeito do comportamento de outro agente **B** conforme as convenções sociais e, as suas intenções em relação a **B**, ocorrem de modo que **B** ocupe uma posição definida pelas expectativas que o agente **A** tem de **B**.

Um estudo categorial sobre estes quatro estados mentais e os seus relacionamentos entre si pode ser visto em Jung [JUN 2000].

2.4.4 Sociedade de Agentes

Coelho afirma que as sociedades de múltiplos agentes surgiram, na informática em geral, quando a proposta dos sistemas abertos (*Open System Interconnection* do *Institute of Standards Organization*) começou a ser aceita por todos [COE 94].

Segundo Sichman [SIC 98], o mecanismo de raciocínio social, baseado na noção de dependência social, é considerado uma construção essencial de agentes realmente

autônomos, imersos no contexto de sistemas multiagentes abertos, isto é, onde agentes podem entrar e sair dinamicamente da sociedade, sem nenhum controle global. O mecanismo social é qualquer mecanismo de raciocínio que usa informação sobre os outros em ordem para inferir alguma conclusão, segundo Sichman et al. [SIC 94]. Conseqüentemente, a existência de um mecanismo assim dentro de um agente significa:

- um agente precisa explicitamente representar algumas propriedades dos outros, que podem mudar dinamicamente;
- um agente precisa explorar esta representação, e assim otimizar seu comportamento de acordo com a evolução da sociedade;
- um agente precisa revisar suas representações quando ele detecta que suas crenças sobre os outros podem estar incorretas ou incompletas. Esta revisão precisa ser feita de uma maneira autônoma, sem controle global preestabelecido.

Assim como a adaptação de um agente num cenário que interessa, as relações de dependência permitem um agente conhecer que suas metas são alcançáveis e que seus planos são possíveis em qualquer momento.

Deste modo, um agente pode dinamicamente escolher uma meta para perseguir e um plano para alcançar, estando certo de que todas as habilidades necessárias para executar o plano selecionado estão disponíveis na sociedade.

Interessando formação de coalizão (união), introduz-se a noção de situação de dependência, que permite um agente avaliar a susceptibilidade de outros agentes adotarem suas metas, desde que agentes não estejam necessariamente supostos a serem benevolentes e assim automaticamente adotam as metas de cada outro.

Finalmente, na revisão de consideração de crença, o mecanismo de raciocínio social permite um agente detectar que sua representação de outros está inconsistente. Devido as interações dos agentes serem guiadas por suas informações sobre os outros, é exatamente durante esta interação que eles podem detectar que sua informação está incorreta ou incompleta, e eventualmente revisar isto.

Em Conte & Castelfranchi [CON 92], duas diferentes aproximações para modelagem das interações sociais são apresentadas:

- modelos top-down: nestes modelos, agentes são considerados a ter um problema global a ser resolvido “a priori”. Por isso, cooperação é tomada para concessão. Interações sociais são usualmente obrigadas por algumas estruturas organizacionais preestabelecidas, que “guiam” agentes em ordem para alcançar a meta global. Estes modelos são freqüentemente usados sem sistemas multiagentes que adotam a perspectiva de resolução de problemas;
- modelo bottom-up: nestes modelos, agentes não têm metas comuns “a priori”. Interações sociais são produzidas como um resultado de seus esforços para alcançar suas próprias metas. Nenhuma cooperação ou qualquer outro tipo de estrutura organizacional é preestabelecida no início. Estes modelos são usados na maioria dos SMA que adotam perspectiva de simulação social.

A noção de estrutura organizacional que é dinamicamente construída pelos agentes nos modelos bottom-up é chamada de coalizão. Isto é mostrado em Conte & Sichman [CON 95] que estes modelos podem ser classificados em dois pontos principais:

- modelos baseados na utilidade: nestes, como na teoria dos jogos, em que é considerado as interações sociais entre os agentes, onde o último precisa coordenar a si próprio para o seu ser um comportamento global coerente. Isto é usualmente realizado pelo desenvolvimento de convenções e obrigações. A

existência de outros agentes limita a autonomia, capacidade e realização dos agentes individuais;

- modelos baseados na complementaridade: estes propõem uma perspectiva diferente nas interações sociais, levando em conta o fato que agentes podem ter habilidades complementares, que podem ser necessárias para alcançar as metas dos próprios agentes. Dessa maneira, a existência de outros agentes aumenta a autonomia e capacidade dos agentes individuais. Igualmente, se um agente não pôde alcançar alguma meta em si próprio, ele pode alcançar pela solicitação de outros agentes para ajudar.

O mecanismo de raciocínio social está baseado na noção de dependência social. Em suma, um agente é dito ser dependente de outro se o último pode facilitar/preveni-lo do alcance de sua meta. A noção de dependência é incerta para a capacidade social segundo Castelfranchi [CAS 90]: se um agente é dependente de outro, o último ganha o poder sobre ele. Relações de dependência podem ser unilateral ou bilateral. A respeito da relação bilateral, dependência mútua é o caso quando dois agentes dependem um do outro para a mesma meta, enquanto dependência recíproca é o caso quando dois agentes dependem um do outro, mas para metas diferentes. Dependência mútua conduz à cooperação, enquanto que dependência recíproca conduz à troca social.

Alguns aspectos essenciais do mecanismo de raciocínio social dos agentes:

- adaptação de um agente - a relação de dependência permite um agente conhecer que suas metas são alcançáveis e que seus planos são executáveis em qualquer momento. Um plano é dito ser executável se todas as ações necessárias para concluir este plano pode ser executada por no mínimo um membro corrente da sociedade. Uma meta alcançável é uma meta que tem no mínimo um plano executável que quando completado alcança esta meta. Como resultado, um agente pode usar seu mecanismo de raciocínio social para escolher dinamicamente uma meta para perseguir e um plano para alcançá-la, estando certo que todas habilidades necessárias para concluir o plano selecionado estão disponíveis na sociedade;
- formação de coalizão - introduz-se a noção de situação de dependência, que permite um agente avaliar a susceptibilidade de outros agentes adotarem suas metas. Sendo este o caso, a escolha do sócio é feita mais eficientemente, porque agentes não necessariamente supõem ser benevolentes, isto é, eles não adotam automaticamente as metas dos outros;
- revisão de crenças - o mecanismo de raciocínio social permite um agente detectar que sua representação dos outros está inconsistente. Devido as interações dos agentes serem guiadas por suas informações sobre os outros, é durante interações que eles podem detectar que estas informações estão incorretas ou incompletas. Nos casos gerais de SMA abertos, a completa e total informação correta sobre cada outro é claramente uma exceção, porque agentes têm somente descrições parciais de um outro e de seu ambiente. Pela detecção incorreta/incompleta que acredita ter sobre outros, um agente pode revisá-los para restaurar a consistência.

Princípios Básicos

Sichman & Conte [SIC 98a] apresentam alguns princípios que os agentes devem ter na sociedade:

- Princípio da não benevolência: agentes não são presumidos a ajudar outro agente, isto é, eles decidem autonomamente quando ou não cooperar com outros;
- Princípio da sinceridade: agentes não tentam aproveitar-se de outro agente, ou seja, eles não oferecem informações errôneas deliberadamente e sempre comunicam informações em que eles acreditam;
- Princípio do conhecimento próprio: agentes têm uma completa e correta representação de si mesmos (suas metas, suas perícias). Entretanto, agentes podem ter crenças sobre outros que são incorretas ou incompletas;
- Princípio da consistência: agentes não mantêm crenças contraditórias sobre os outros. Quando uma inconsistência é encontrada, eles revisam suas crenças em ordem para restabelecer um estado consistente.

Mecanismo de Raciocínio Social

A seguir, apresenta-se rapidamente o núcleo do mecanismo do raciocínio social, segundo Sichman et al. [SIC 94], e Sichman & Conte [SIC 98a]:

- Descrição Externa: chamamos de descrição externa a representação que um agente tem sobre os outros. Esta representação é particular, que é adquirida de diferentes fontes de informação: percepção, comunicação e inferência. Uma descrição externa consiste de várias entradas, cada uma contendo alguma informação sobre um agente em particular. Uma entrada de descrição externa contém as metas que o agente está tentando alcançar, as ações que ele é capaz de executar, os recursos sobre o qual ele tem controle e os planos que ele tem na ordem para alcançar suas metas. Um plano é uma seqüência de ações instanciadas, composta de uma simples ação mais uma coleção de recursos necessários para executá-lo.
- Rede de Dependência: um agente armazena todas as suas relações de dependência numa estrutura simples chamada rede de dependência.
- Situação da meta: relata um agente para uma certa meta. As 4 possíveis situações de meta são:
    - sem meta (*no goal* - NG): o agente não obteve a meta em sua lista de metas;
    - sem plano (*no plans* - NP): o agente obteve a meta em sua lista de metas, mas ele não obteve nenhum plano para executá-la;
    - autônomo (AUT): o agente obteve a meta e alguns planos para executá-la. Além disso, no mínimo algum destes planos é certo que ele pode executar todas as ações necessárias por si próprio, sem necessitar pedir ajuda para outros agentes;
    - dependente (DEP): o agente obteve a meta e alguns planos para executá-la. Entretanto, ele não pode executar todas as ações necessárias nele próprio em qualquer destes planos.
- Situação de Dependência: se um agente depende de outro para uma certa meta, ele pode desejar calcular se o último também depende dele para alguma de suas metas. Sendo o caso, uma proposta de cooperação ou troca social pode ser enviada para este agente, com uma chance de aceitação. Por outro lado, os agentes são heterogêneos, e seus mecanismos de planejamento podem diferir. Um agente pode assim inferir uma dependência mútua em

relação a ele e um possível sócio, enquanto que o último não infere a mesma dependência bilateral. Considerando dois agentes *i* e *j*, onde o agente raciocinando é *i*. Se *i* infere a situação de meta DEP para a meta *g*, seis diferentes situações de dependência entre ele e *j* podem ocorrer:

1. Independência (IND): usando seu próprio plano, ele infere que não depende de *j* para a meta *g*;
2. dependência mútua acreditada localmente (LBMD- *locally believed mutual dependence*): usando seu próprio plano, ele infere uma dependência mútua entre ele e *j* para a meta *g*, mas ele não infere a mesma conclusão usando os planos que ele acredita que *j* tenha obtido;
3. dependência mútua acreditada mutuamente (MBMD - *mutually believed mutual dependence*): usando ambos seus próprios planos e os planos que ele acredita que *j* tenha obtido, ele infere uma dependência mútua entre eles;
4. dependência recíproca acreditada localmente (LBRD - *locally believed reciprocal dependence*): usando seu próprio plano, ele infere uma dependência recíproca entre ele e *j* para a meta *g* e *g'*, mas ele não infere a mesma conclusão usando os planos que ele acredita que *j* tenha obtido;
5. dependência recíproca acreditada mutuamente (MBRD - *mutually believed reciprocal dependence*): usando ambos seus próprios planos e os planos que ele acredita que *j* tenha obtido, ele infere uma dependência recíproca entre eles;
6. dependência unilateral (UD - *unilateral dependence*): usando seu próprio plano, ele infere que ele depende de *j* para a meta *g*, mas o último não depende dele para qualquer de suas metas.

Porém as representações mútuas dos agentes podem estar inconsistentes. Sichman & Demazeau abordam em [SIC 95] como isto pode ser tratado em termos de estarem incompletas ou incorretas as representações mútuas de ações necessárias e ações oferecidas no mecanismo de raciocínio social entre dois agentes.

As noções de meta e situações de dependência são usadas por um agente em ordem para escolher um sócio para o qual uma proposta de coalizão está para ser enviada, quando um agente não alcançar uma meta intencionada por ele próprio. Os estados de critério de escolha de sócio baseado na dependência mútua são:

- uma dependência mútua (acreditada mutuamente ou localmente) é sempre a melhor escolha, desde que o problema da reciprocidade não apareça;
- uma dependência acreditada mutuamente (mútua ou recíproca) é sempre a melhor escolha, desde que o problema do convencimento de outro não apareça.

2.4.5 Comunicação entre Agentes

A comunicação é o processo pelo qual a informação é trocada numa transação do agente, podendo ocorrer entre os agentes, por exemplo, através do ambiente pela troca de mensagens com a utilização de um quadro negro.

Segundo Finin et al. [FIN 93], a troca ou passagem de mensagens entre agentes pode ser feita de três formas distintas:

- **ponto-a-ponto**: nesta abordagem, as mensagens são enviadas para um endereço específico (o receptor) que deve ser conhecido pelo emissor. Como vantagens desta abordagem, têm-se: um agente sempre sabe para onde uma mensagem está sendo emitida; e que os controles de segurança são facilmente introduzidos, já que um agente pode assegurar que a mensagem nunca será enviada para agentes desconhecidos;
- ***broadcast***: a passagem de mensagem *broadcast* é baseada na emissão de uma mensagem não para um endereço específico, mas para todos os agentes da sociedade (usado na estrutura do tipo estrela). Nesta abordagem, um agente particular pode ser substituído por outro agente com conduta equivalente e o procedimento do sistema como um todo será inalterado. A passagem de mensagem *broadcast* não é segura, já que qualquer agente pode examinar os conteúdos de qualquer mensagem;
- ***multicast***: uma maneira que está sendo utilizada para evitar os problemas do *broadcast* é a estruturação do espaço de agentes em grupos (usado na estrutura do tipo hierárquica). Assim, cada agente é membro de um grupo menor e se esse agente transmite uma mensagem, esta será emitida para todos os membros de um ou mais (dependendo do sistema) grupos de agentes.

Linguagens de Comunicação de SMA

A linguagem de comunicação é o meio através do qual atitudes (afirmação, requisição e consulta) a respeito do conteúdo da troca de conhecimento são comunicadas, segundo Finin et al. [FIN 97].

Existem linguagens que são destinadas exclusivamente à comunicação entre agentes, as quais definem padrões para a troca de mensagens. O exemplo mais conhecido atualmente é a ACL (*Agent Communication Language*), que é uma Linguagem de Comunicação de Agentes resultado de um projeto americano financiado pela ARPA (*Advanced Research Projects Agency*), denominado KSE (*Knowledge Sharing Effort*).

Quatro áreas foram identificadas como sendo um esforço inicial, segundo Patil et al. [PAT 97]:

- mecanismo de tradução entre bases de conhecimento representadas em diferentes linguagens;
- versões comuns de linguagem e módulos de raciocínio com famílias de paradigmas representacionais;
- protocolos para comunicação entre módulos de base de conhecimento separadas, assim como sistemas de base de conhecimentos e bases de dados;
- bibliotecas de “ontologias”, isto é, fundações pré-fabricadas para bases de conhecimentos de aplicações específicas numa área de tópico particular.

Assim, este projeto é composto, inicialmente, de quatro partes, podendo ser melhor visualizadas em Patil et al. [PAT 97]:

- um formato para intercâmbio do conhecimento (KIF – *Knowledge Interchange Format*);
- uma especificação para o componente representacional de família de sistemas de representação de conhecimento (KRSS – *Knowledge Representation System Specification*)
- uma linguagem para manipulação (protocolo) e consulta do conhecimento (KQML – *Knowledge Query and Manipulation Language*);

- um formato para ontologias (onto língua).

Segundo Finin et al. [FIN 97], para que agentes interajam e interoperem efetivamente, é necessário três componentes fundamentais e distintos:

- uma linguagem comum;
- um entendimento comum da troca de conhecimento;
- a habilidade de trocar tudo o que está incluído nos dois itens anteriores.

As linguagens de comunicação de agentes possuem algumas características relevantes na determinação de sua adequação, segundo Mayfield et al. [MAY 96] e Finin et al. [FIN 97]:

- forma: uma boa linguagem de comunicação de agentes deve ser declarativa, sintaticamente simples e legível. Como uma linguagem de comunicação deve ser integrada em uma grande variedade de sistemas, sua sintaxe deve ser extensível;
- conteúdo: uma linguagem de comunicação deve ser estendida de modo que se adapte bem com outros sistemas. A linguagem deve fornecer um conjunto definido de ações de comunicação (primitivas);
- semântica: a descrição da semântica de uma linguagem normalmente é feita através da linguagem natural, a linguagem deve ser baseada numa teoria, não deve ser ambígua e deve considerar tempo e local;
- implementação: a implementação deve ser eficiente, tanto para velocidade como para a utilização da largura da banda. Ela deve prover uma boa adaptação com a tecnologia de software existente. A interface deve ser fácil de utilizar; detalhes das camadas de rede que existem abaixo das primitivas de ações de comunicação devem ser escondidas do usuário. A linguagem deve ser apropriada para implementação parcial, para que os agentes inteligentes possam manusear somente um subconjunto de primitivas de ações de comunicação;
- rede: uma linguagem de comunicação de agente deve adaptar-se bem com as tecnologias modernas de rede. A linguagem deve suportar todas conexões básicas - ponto-a-ponto, *broadcast* e *multicast*. As conexões síncrona e assíncrona devem ser suportadas. A linguagem deve conter um conjunto de primitivas bastante ricas que possam servir como um substrato sobre quais linguagens de alto-nível e protocolos podem ser desenvolvidos. Além disso, estes protocolos de alto-nível devem ser independentes dos mecanismos de transporte utilizados (por exemplo, TCP/IP, e-mail, HTTP, etc.);
- ambiente: o ambiente em que agentes inteligentes devem trabalhar deve ser altamente distribuído, heterogêneo e extremamente dinâmico. Ele deve suportar interoperabilidade com outras linguagens e protocolos;
- confiabilidade: a linguagem de comunicação deve suportar uma comunicação confiável e segura entre agentes. Devem ser oferecidos recursos para trocas privadas e seguras entre dois agentes. A linguagem deve suportar mecanismos razoáveis para identificação e sinalização de erros e advertências.

KQML (*Knowledge Query and Manipulation Language*)

Uma das linguagens de comunicação entre agentes que procura implementar as características acima citadas é KQML que é uma linguagem e um protocolo para troca de informação e conhecimento entre agentes.

KQML tem por objetivo desenvolver técnicas e metodologias para a construção de bases de conhecimento de larga escala que são compartilháveis e reutilizáveis. KQML pode ser utilizado como a linguagem que um programa aplicativo utiliza para interagir com um sistema inteligente, ou para dois ou mais sistemas inteligentes compartilharem informação e conhecimento para solução cooperativa de problemas, segundo Hübner & Sichman [HÜB 2000].

Algumas características importantes de KQML citadas por Hübner & Sichman [HÜB 2000]:

- qualquer linguagem pode ser usada para escrever o conteúdo das mensagens (Lisp, Prolog, Português, SQL - *Structured Query Language*);
- as informações necessárias para compreender o conteúdo das mensagens estão incluídas na própria comunicação;
- quando os agentes trocam mensagens KQML, o mecanismo de transporte é transparente, isto é, em termos da mensagem sair do agente emissor e chegar no receptor;
- o formato das mensagens é simples, fácil de ler por pessoas e de ser analisada por um *parser*.

A estrutura da mensagem KQML sugerida por Patil et al. [PAT 97] pode ser visualizada na FIGURA 2.2:

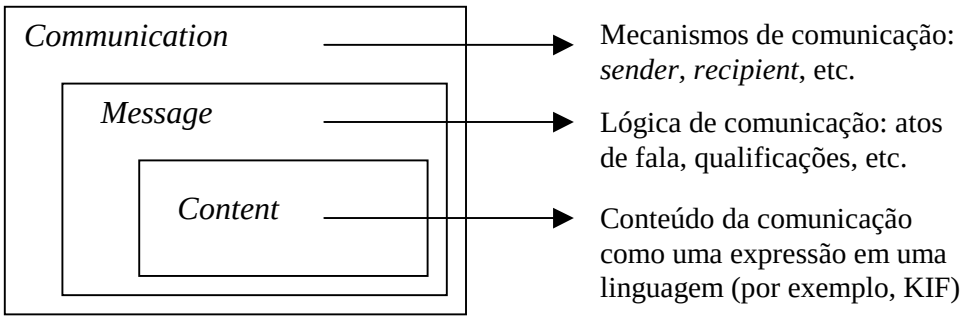

FIGURA 2.2 – Estrutura de uma mensagem KQML

Esta figura representa a estrutura de uma mensagem KQML composta pelos mecanismos de comunicação, pela mensagem representada através da lógica de comunicação, e pelo conteúdo da mensagem.

Com a finalidade de definir uma forma de cooperação entre os agentes, foi elaborado um protocolo de troca de mensagens. Tal comunicação entre os agentes ocorre através de uma definição de mensagem baseada em KQML, descrita, mais adiante, na seção Mensagens, buscando assim, construir uma arquitetura de agentes robusta e padronizada o quanto for possível e que permita a reutilização de códigos para os diferentes tipos de agentes.

KQML é indiferente para o formato da própria informação, assim, KQML contém subexpressões chamadas conteúdo das mensagens, segundo Finin et al. [FIN 93]. As mensagens são definidas em duas partes, explicitadas, mais adiante, nas seções Implementação KQML e Conteúdo das Mensagens. São elas:

- cabeçalho da mensagem: segue o padrão KQML;
- conteúdo da mensagem: pode assumir variados formatos, dependendo da situação na qual o agente se encontra.

Assim, os agentes terão a capacidade de decompor as mensagens e decidir quais as ações pessoais e de comunicação que devem executar.

KQML é uma linguagem de alto nível usada por sistemas baseados em conhecimento para compartilhar conhecimento e informação em tempo de execução. KQML é uma linguagem externa que foca principalmente o formato das mensagens. Em KQML, formatos de mensagens são definidos, sendo que pode ser usada uma outra linguagem para expressar o conteúdo da mensagem. As mensagens KQML são chamadas performativas e estão baseadas na Teoria de Atos de Fala. Cada mensagem é enviada com o objetivo de gerar uma determinada ação, segundo Finin et al. [FIN 93].

Esta linguagem enfoca um grande conjunto de mensagens predefinidas (performativas), as quais definem as operações possíveis de serem executadas pelos agentes. Assim, aqueles que estão em conformidade com a KQML podem responder a estas mensagens de maneira adequada independentemente da estrutura do seu emissor.

Esta tecnologia se torna importante em virtude da crescente necessidade de uma linguagem comum entre agentes, principalmente numa rede de grande magnitude como a Internet. Um agente, ao receber uma mensagem KQML, toma uma ação com base no seu significado e conteúdo, segundo Mayfield et al. [MAY 96].

KQML é uma linguagem projetada para suportar interações entre agentes inteligentes. Ela foi desenvolvida pela ARPA no Programa de Compartilhamento de Informações (*Knowledge Sharing Effort*) e implementada independentemente por vários grupos de pesquisa. Foi utilizada com sucesso para implementar uma variedade de sistemas de informações usando diferentes arquiteturas de software, segundo Mayfield et al. [MAY 96].

Na especificação da KQML, os agentes que participam através da troca de mensagens podem ser comparados com entidades que possuem comportamento próprio e autônomo, agindo em sociedade através da interação com outros agentes. Esse comportamento está intimamente ligado ao seu estado mental que pode ser representado por uma base de conhecimento onde informações que descrevem suas crenças e objetivos estão armazenadas.

As crenças do agente representam as informações sobre si próprio e sobre o ambiente externo, incluindo as bases de conhecimentos de outras entidades. Todas as performativas giram em torno das bases de conhecimentos, ou seja, as trocas de mensagens estão normalmente associadas com as crenças ou objetivos contidos na base do agente chamada de Virtual Knowledge Base (VKB), de forma que sua implementação não deve necessariamente ser estruturada como uma base de conhecimento, segundo Finin et al. [FIN 93].

O conjunto de mensagens KQML pode ser estendido desde que as novas performativas criadas obedeçam a mesma forma da especificação original da linguagem. Os agentes que estão de acordo com a KQML não precisam reconhecer todas as mensagens, de forma que um pequeno subconjunto pode ser suficiente num determinado sistema multiagente, ou seja, dependendo da necessidade, pode-se escolher somente algumas performativas a serem utilizadas na comunicação.

Uma mensagem KQML, também chamada de performativa, é expressa como uma *string* de caracteres ASCII usando a sintaxe definida por Finin et al. [FIN 93], constituída de palavras chaves (chamadas nomes de parâmetros), devem começar por dois pontos ":" e precedem um valor do parâmetro correspondente.

Agentes

Os agentes são implementados na classe Agent, como mostra a FIGURA 2.3:

Classe Agent		
Parâmetros:		
	name	(nome do agente)
	description	(descrição do agente)
	address	(endereço na rede)
	port	(porta de comunicação)

FIGURA 2.3 – Classe Agent

Esta figura representa a implementação do agente identificado pelos parâmetros: nome, descrição, endereço e porta.

Mensagens

As mensagens e a troca destas entre os agentes foram baseadas numa das principais contribuições do trabalho feito por Bica [BIC 99]. A implementação da mensagem, é composta por dois módulos:

- um formato geral que possui especificação KQML (classe KQMLMessage);
- um formato específico que compõe a tarefa que o agente deve executar (classe ContentMessage).

A especificação da mensagem foi elaborada desta maneira pois o RMI (*Remote Method Invocation*) permite a passagem de objetos criados pelo implementador.

Implementação KQML

KQML possui um conjunto de parâmetros chaves reservados, sendo útil para estabelecer uma uniformidade dos parâmetros e suporte para que programas entendam performativas desconhecidas mas com parâmetros conhecidos. Os parâmetros reservados e seus significados estão demonstrados na TABELA 2.1, segundo Finin et al. [FIN 93]:

TABELA 2.1 - Parâmetros de KQML

Parâmetros	Significado
:sender	o atual transmissor de uma performativa
:receiver	o atual receptor da performativa
:language	o nome da linguagem de representação do parâmetro :content
:ontology	o nome da ontologia usada no :content
:in-reply-to	a indicação esperada na resposta
:reply-with	quando um emissor espera uma resposta, e se sim, um rótulo para a resposta
:force	se o transmissor irá sempre negar o significado da performativa
:content	a informação sobre o que a performativa expressa

Tais parâmetros foram implementados na forma de uma classe Java, chamada KQMLMessage, como mostra a FIGURA 2.4:

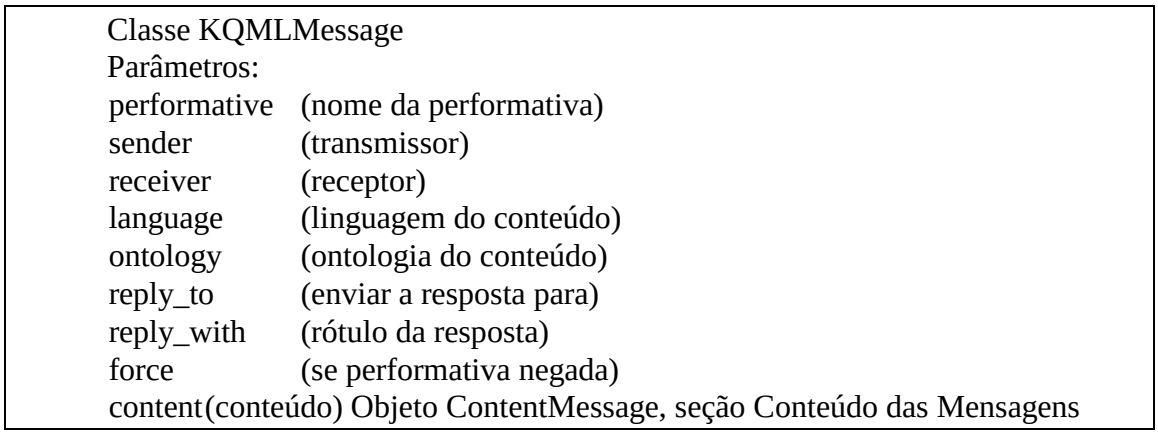

FIGURA 2.4 – Classe KQMLMessage

A TABELA 2.2, identifica alguns exemplos de performativas KQML. `S` significa o remetente e `R` o receptor. A lista completa e detalhada de performativas KQML pode ser encontrada em Finin et al. [FIN 93].

TABELA 2.2 - Exemplos de Performativas KQML

Nome	**Resposta requerida**	**Significado**
Achieve		S deseja que R torne algo verdadeiro em seu ambiente
Break		S deseja que R rompa um caminho estabelecido
Delete_all		S deseja que R remova todas as sentenças que se encontram na BCV (Base de Conhecimento Virtual)
Delete_one		S deseja que R remova uma sentença da BCV
Error		S considera uma mensagem de R mal formulada
Forward		S deseja que R repasse a mensagem para outro receptor
Insert		S pergunta a R para adicionar o conteúdo (:content) na BCV
Sorry		S não pode mais prover informações para o reply da performativa
Tell	X	a sentença está na BCV do S

Conteúdo das Mensagens

Seguindo o formato de mensagem KQML, é necessário especificar o parâmetro *content*. Para isso, foi implementada a classe ContentMessage, como mostra a FIGURA 2.5, que possui todos os parâmetros necessários para compor uma tarefa. Tais parâmetros são utilizados conforme o agente e o tipo de mensagem que este deseja enviar, podendo ser melhor visualizado na seção 5.1.

Classe ContentMessage	
Parâmetros:	
subject	(matéria atual)
unit	(código do capítulo)
user	(usuário que requisitou)
password	(senha do usuário)
content	(conteúdo a ser mostrado)
pattern	(padrão a ser reconhecido)
response	(resposta do exercício/reconhecimento do padrão)
operation	(operação realizada pelo usuário, exemplo: anterior/próximo)
type	(tipo da mensagem)
priority	(prioridade da mensagem)
performance	(desempenho/nível do aluno)
preference	(preferência do usuário)
tatic	(tática pedagógica)

FIGURA 2.5 – Classe ContentMessage

Java

Além da independência de plataforma e das fortes capacidades de rede, Java oferece os benefícios da orientação a objetos (OO) e de múltiplas linhas de execução (*multithreading*). A linguagem também é dinâmica: pequenas partes do código Java são montados em tempo de execução (*runtime*) dentro do programa, segundo Deitel & Deitel [DEI 2001].

Atualmente, na Internet, muito se fala sobre o Java; grande parte desta agitação foi gerada pelos *applets* que são pequenos programas que podem ser embutidos em páginas Web. A execução dos *applets* no lado cliente é uma grande inovação na programação Web.

A linguagem Java foi projetada para ser segura no sentido de que o seu código não prejudicaria a si mesmo ou a outros componentes de software. Por exemplo, não é permitido a manipulação explícita de ponteiros.

Antes que algum código de classe Java seja executado, seus *bytecodes* são verificados. O verificador de *bytecode* utiliza um teorema simples para assegurar que:

- ponteiros não foram esquecidos;
- restrições de acesso não foram violadas;
- as chamadas dos métodos contêm o número e tipo corretos de parâmetros;
- a pilha não está em *overflow*.

Este passo de verificação é muito desejável do ponto de vista da segurança, e tem adicionado benefícios para que o interpretador execute mais rapidamente.

Os programas Java rodam dentro de máquinas virtuais, as quais ficam dentro do computador no qual eles estão rodando. A máquina virtual age como uma parede entre o *host* e o programa Java. Um programa Java nunca acessa os dispositivos de entrada e saída, o sistema de arquivos, ou mesmo a memória do computador no qual está rodando. Em vez disso, ele pede que a máquina virtual os acesse. A máquina virtual contém ferramentas de manipulação de *strings*, rotinas gráficas e de interface com o usuário, estruturas básicas de dados e funcionalidades matemáticas. Para utilizá-los, é necessário aprender sobre a Interface de Programação de Aplicativos (API - *Application Program Interface*), que é uma coleção de classes, interfaces e exceções prontas. Dentro da API,

essas classes, interfaces e exceções estão agrupadas em oito pacotes, segundo Deitel & Deitel [DEI 2001]:

- java.applet - contém classes e interfaces que ativam as *applets*;
- java.awt - permite escrever interfaces gráficas de usuário;
- java.awt.image - dedicado à criação e manipulação de imagens;
- java.awt.peer - composto totalmente de interfaces que permitem que o sistema de janelas do Java seja facilmente portado entre plataformas;
- java.io - trata a entrada e saída do programa;
- java.lang - contém os elementos centrais da linguagem Java, tais como *Object*, *String*, *Exception* e *Thread*;
- java.net - trata a interação com a rede;
- java.util - contém várias classes de utilitários que tornam mais fácil a programação.

Sockets

Uma forma dos programas Java interagirem com outros programas é a utilização de *sockets*.

Um *socket* é uma conexão de dados transparente entre dois computadores numa rede. O pacote java.net provê duas classes - Socket e ServerSocket - que implementam, respectivamente, o lado da conexão do cliente e o lado do servidor.

Um *socket* é identificado pelo endereço de rede e por um número de porta do computador, só assim, a camada TCP (*Transmission Control Protocol*) poderá identificar a aplicação na qual os dados serão transmitidos.

RMI (*Remote Method Invocation*)

Além dos *sockets*, existe outra forma dos programas Java interagirem uns com os outros: a utilização do RMI [SUN 98a]. RMI provê um modelo simples e direto para computação distribuída com objetos Java. Para utilizá-lo, é necessário implementar um Servidor com todos os métodos necessários para a troca de informações e os Clientes que irão acessá-lo.

Quando um cliente deseja enviar uma mensagem ao servidor, ele invoca um método do servidor e o mesmo se dá quando o desejo é o de receber uma mensagem.

RMI pode passar objetos como argumento e retornar valores de tipos de dados não predefinidos, portanto, os objetos podem fazer parte da API ou serem criados pelo implementador.

O sistema de RMI consiste em três camadas: a camada de *stub/skeleton*, a camada de referência remota e a camada de transporte. Cada camada é definida por uma interface específica e protocolo, portanto cada uma delas é independente da próxima e pode ser substituída por diferentes tipos de implementação sem afetar as outras camadas no sistema.

JDBC (*Java Data Base Connection*)

O JDBC consiste em uma API Java, composto por um conjunto de classes e interfaces escritas na linguagem Java e que são responsáveis pela execução de operações SQL, segundo Deitel & Deitel [DEI 2001]. O JDBC provê uma forma simples de

desenvolver aplicações com banco de dados utilizando a linguagem Java, sendo três as suas funções:

- estabelecer a conexão com o banco de dados;
- enviar ao banco consultas SQL;
- processar os resultados das consultas.

O banco de dados utilizado na implementação pode ser o banco de dados Oracle. A Oracle [ORA 99] desenvolveu drivers que possuem extensões de propriedades, tipos e performance, melhorando assim o desempenho do JDBC fornecido pelo Java. Os drivers existentes são: JDBC Thin (utilizado em *Applets*) e o JDBC OCI (utilizado em aplicações cliente-servidor).

Ambiente de Desenvolvimento – JAT (*Java Agent Template*)

O JAT [FRO 98], desenvolvido por Robert Frost na Universidade de Stanford, fornece um conjunto de pacotes escrito na linguagem Java que permite a construção de agentes de software os quais se comunicam com uma comunidade de outros agentes distribuídos na Internet. O modelo foi todo construído na linguagem Java, que é portável, porém os agentes projetados no JAT não são migratórios e, portanto, possuem uma existência fixa em um determinado *host*. Esses agentes podem ser executados como *applets* Java e como aplicações *stand-alone*, sendo que ambas as configurações suportam agentes gráficos (possuem uma interface para comunicação com o usuário) ou não gráficos.

A coordenação na sociedade é realizada por um agente servidor de nomes (ANS) dedicado à manutenção dinâmica, denominação única dos agentes e endereçamento. O ANS deve estar continuamente rodando durante toda a vida do grupo de agentes e irá manter um registro dos nomes e endereços Internet de todos os agentes.

JAT pode ser usado como plataforma para construção de agentes para aplicações de diferentes domínios. Os agentes são criados através da especialização das classes Java fornecidas pelo ambiente.

2.5 Engenharia Semiótica

Umberto Eco afirma que a semiótica é a disciplina que estuda os signos, sistemas de signos, significação, comunicação e todos os processos culturais [ECO 80].

Por processos culturais, Eco entende como sendo a cultura que continuamente traduz signos em outros signos, definições em outras definições, e assim por diante, propondo, dessa forma, uma cadeia ininterrupta de unidades culturais que compõem outras unidades culturais [ECO 80]. Podemos citar, como exemplo, o comportamento padrão de um grupo de soldados que interpreta o sinal de “atenção” tocado por uma corneta, dando, assim, uma informação a respeito da unidade cultural (no caso um comando) veiculada pelo significante musical.

A semiótica de Charles Sanders Peirce [PEI 2000] destaca a importância do papel dos signos no raciocínio humano. Para Peirce, semiótica é lógica, ou seja, é a teoria das condições gerais da referência dos Símbolos e outros Signos aos seus Objetos manifestos (ou Interpretantes que pretendem determinar), sendo assim a teoria das condições da verdade.

A abordagem da Engenharia Semiótica apresenta, para IHC (Interação Homem-Computador), uma perspectiva na qual o sistema computacional é um artefato de *metacomunicação* e através dele o *designer* envia uma mensagem para os usuários, cujo

conteúdo deve ser o modelo de interação e de funcionalidade do sistema, segundo Souza [SOU 93]. O conteúdo de tal mensagem é o modelo de usabilidade da aplicação. A sua expressão é formada pelo conjunto de todas a mensagens veiculadas na interface durante o processo de interação. O usuário exerce o duplo papel de interagir com o sistema e interpretar uma mensagem enviada pelo *designer*.

Uma característica da mensagem do *designer* é que ela é dinâmica, ou seja, varia ao longo do tempo, revelando variações de significado. Este é o caso dos objetos de interfaces indicadores de estado ou comportamento de um sistema. Mensagens desse tipo se comportam como agentes ativos e são chamadas de *performáticas*. Outro caso de mensagem dinâmica é quando ela pode ser construída interativamente pelo usuário, possibilitando e incentivando ações do usuário. Temos como exemplo, as cascatas de menus que vão sendo construídas durante o processo de interação.

A Engenharia Semiótica surgiu com o objetivo de subsidiar soluções práticas para o *design* de interfaces de usuário, cujo autor legítimo das informações ou mensagens é justamente o *designer*. Este aspecto é a premissa básica da Engenharia Semiótica.

A Engenharia Semiótica argumenta que o desafio de usabilidade pode ser resolvido com a perspectiva de que *designers* devem projetar e comunicar uma aplicação de software cuja interface seja vista como um sistema de comunicação para o usuário interagir e como um *medium* que (meta) comunica sua funcionalidade e o próprio sistema de comunicação, segundo Souza [SOU 93]. Para colocarmos em prática esta perspectiva, é preciso considerar uma hipótese semiótica fundamental apresentada por Peirce: a de que signos e sistemas semióticos são a ferramenta intelectual mais poderosa que as pessoas têm para a aquisição e comunicação de conhecimento [PEI 2000].

O objetivo do sistema semiótico é apoiar a elaboração da mensagem do *designer* a ser veiculada através do sistema, oferecendo recursos para a elaboração da sua expressão e do seu conteúdo. Ele deve orientar o *designer* na escolha dos elementos expressivos da interface (como os *widgets*[3], por exemplo, que são os componentes visuais interativos) que permitam ao usuário interpretar as funções, os objetos e os modos de interação com o sistema. Assim, ele contribui na facilidade de aprendizado do sistema através de uma interface que comunica o modelo de usabilidade que o *designer* concebeu.

2.5.1 Os Signos

Um ***signo*** é definido por Peirce [PEI 2000] como:

"Um signo, ou *representâmen*, é aquilo que, sob certo aspecto ou modo, representa algo para alguém. Dirige-se a alguém, isto é, cria na mente dessa pessoa um signo equivalente, ou talvez um signo mais desenvolvido. Ao signo assim criado denomino *interpretante* do primeiro signo". [Op. Cit. (pág.46)]

"Qualquer coisa que conduz uma outra coisa (seu *interpretante*) a referir-se a um objeto ao qual ela mesma se refere (seu *objeto*), de modo idêntico, transformando-se o interpretante, por sua vez, em signo, e assim sucessivamente *ad infinitum*". [Op. Cit. (pág. 74)]

[3] os *widgets* são, por exemplo, os botões de comando, botões de opções, rótulos, caixas de texto, caixas de diálogos.

"Um *Signo*, ou *Representâmen*, é um Primeiro que se coloca numa relação triádica genuína tal com um Segundo, denominado seu *Objeto*, que é capaz de determinar um Terceiro, denominado seu *Interpretante*, que assuma a mesma relação triádica com seu Objeto na qual ele próprio está em relação com o mesmo Objeto". [Op. Cit. (pág 63)]

"Um *Signo* é tudo aquilo que está relacionado com uma Segunda coisa, seu *objeto*, com respeito a uma Qualidade, de modo tal a trazer uma Terceira coisa, seu *Interpretante*, para uma relação com o mesmo Objeto, e de modo tal a trazer uma Quarta para uma relação com aquele Objeto na mesma forma, *ad infinitum*". [Op. Cit. (pág 28)]

Esta relação triádica existente nas definições de signo dada por Peirce, pode ser melhor resumida pela FIGURA 2.6:

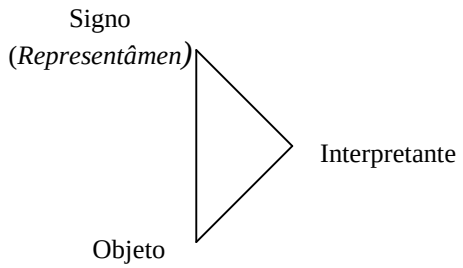

FIGURA 2.6 – O Signo de Peirce

O signo representa alguma coisa (*representâmen*[4]), seu *objeto*, podendo ter mais de um objeto, como por exemplo, a frase "Caim matou Abel", que é um Signo, referindo-se, no mínimo a dois Objetos (Caim e Abel), além de podermos considerar um terceiro Objeto ("um assassinato"), tendo-se, assim, um conjunto de objetos como sendo um Objeto complexo (um Signo mais amplo).

Segundo Peirce [PEI 2000], existem três tipos de signos indispensáveis ao raciocínio, que juntos, formam uma tríade importante:

- o *ícone*, ou signo diagramático, que ostenta uma semelhança ou analogia com o sujeito do discurso;
- o *índice*, que assim como um pronome demonstrativo ou relativo, atrai a atenção para o objeto particular que estamos visando, porém sem descrevê-lo;
- o *símbolo*, que é o nome geral ou descrição que o seu objeto significa, através de uma associação de idéias ou conexão habitual entre o nome e o caráter significado.

Assim, explicam-se o porquê da necessidade de haver três classes de signos, pois existe uma conexão tripla de *signo, coisa significada e cognição produzida na mente*. Pode existir apenas uma relação de razão entre o signo e a coisa significada, onde, neste caso, o signo é um *ícone*. Ou pode existir uma ligação física direta, onde, dessa forma, o signo é um *índice*. Ou, ainda, pode existir uma relação que consiste no fato da mente associar o signo com seu objeto, onde então, o signo é um *nome* (ou *símbolo*).

O ***Ícone*** é um Signo cuja virtude significante se deve apenas à sua Qualidade, por exemplo, quando dizemos que um retrato de uma pessoa que não vimos é *convincente*, onde, a partir da imagem que podemos ver no retrato, somos levados a formar uma idéia da pessoa que o retrato representa. Então, neste caso, o retrato é um

[4] *Representâmen* – quer dizer aquilo que representa.

Ícone. Assim, a maneira de se comunicar diretamente uma idéia, é através de um ícone, podendo ser imagens, diagramas (representam através de relações), ou metáforas (representam através de um paralelismo com alguma outra coisa).

O Ícone não tem conexão dinâmica alguma com o objeto que representa, pois simplesmente acontece que suas qualidades se assemelham às do objeto e excitam sensações análogas na mente para a qual é uma semelhança.

O ***Índice*** é um Signo cuja significação de seu Objeto se deve ao fato dele ter uma Relação genuína com aquele Objeto, sem, no entanto, levar em consideração o Interpretante. Por exemplo, quando ocorre uma batida na porta, temos a indicação de uma visita. Supõe-se que dois amigos se encontrem na estrada e que um deles diga ao outro: "A chaminé daquela casa está acesa". O outro olha a sua volta e vê uma casa com cortina branca e com uma chaminé da qual sai fumaça. Anda mais um pouco e encontra um segundo amigo. Com simplicidade diz: "A chaminé daquela casa está acesa". "Que casa?", pergunta o outro. "Oh, uma casa com cortina branca", responde ele. "Onde está a casa?" pergunta o amigo. Ele deseja um *índice* que ligue a informação que lhe dão com a casa pretendida, pois apenas palavras não podem fazê-lo.

Por fim, o ***Símbolo*** é um Signo cuja virtude significante se deve a um caráter que só pode ser compreendido juntamente com a ajuda de seu Interpretante, onde temos, por exemplo, quando da emissão de um discurso cujos sons e palavras vão determinar, na mente do ouvinte, signos correspondentes.

O *Símbolo* é um *Representâmen* cujo *caráter representativo* consiste exatamente em ser uma regra que *determinará seu Interpretante*. Todas as palavras, frases, livros e outros signos convencionais são símbolos.

Um constituinte de um Símbolo pode ser um Índice, e um outro constituinte pode ser um Ícone. Por exemplo, um homem, que caminha com uma criança, levanta o braço para o ar, aponta e diz: "Lá está um balão". O braço que aponta é uma parte essencial do símbolo, sem a qual este não veicularia nenhuma informação. Mas, se a criança perguntar: "O que é um balão?", e o homem responder: "É algo como uma grande bolha de sabão", ele torna a imagem como sendo uma parte do símbolo.

O fato de todo signo determinar um Interpretante, que também é um signo, tem, como conseqüência, signos justapondo-se a signos. Além disso, o Interpretante não é necessário que realmente exista, basta que seja um ser representando o futuro.

Na definição de signo de Peirce [PEI 2000], se a série de interpretantes sucessivos vem a ter fim, em virtude desse fato, o signo torna-se pelo menos imperfeito. Por exemplo, um índice é um signo que de repente perderia seu caráter que o torna um signo se seu objeto fosse removido, mas que não perderia esse caráter se não houvesse interpretante. Tal é, por exemplo, o caso de um molde com um buraco de projétil como signo de um tiro, pois sem o tiro não teria havido o buraco; porém, nele existe um buraco, quer tenha alguém ou não a capacidade de atribuí-lo a um tiro. Da mesma forma, um *símbolo* perderia o caráter que o torna um signo se não houvesse um interpretante. Tal é o caso de qualquer elocução de discurso que significa aquilo que significa apenas por força de compreender-se que possui essa significação.

O signo não deve ser visto como uma entidade ou elemento, mas como um *processo*. Algo que requer a presença de alguém e que na mente desta pessoa desencadeia interpretantes, num processo ilimitado ao qual Peirce [PEI 2000] denominou ***semiose ilimitada***. As conseqüências destes conceitos são fundamentais na Engenharia Semiótica, pois avalizam a perspectiva de metacomunicação.

Utilizando o conceito de signo, podemos dizer que comunicação é o processo que ocorre entre duas ou mais pessoas quando uma delas produz algo (uma expressão)

que conduz os interpretantes de todas as outras a referir-se a um objeto comum. Comunicação é, portanto, o que ocorre quando alguém produz algo que é, para si próprio, um signo (A) e é, também, para uma outra pessoa, um outro signo (B) cuja expressão e objeto convergem em (A). Os signos são distintos porque os interpretantes não são os mesmos, uma vez que eles são processos ilimitados que ocorrem nas mentes de cada um dos agentes. A expressão e o objeto são elementos que são postos em comum. Chamaremos o primeiro agente de *produtor* - do signo (A) - enquanto que o segundo chamamos de *intérprete*. Utilizando termos mais convencionais, o produtor pode ser chamado de *emissor*, o intérprete de *destinatário*, e o signo de *mensagem*.

De acordo com a teoria semiótica que estamos considerando, para que a comunicação *designer*-usuário seja possível, é preciso considerar que aplicações de software são signos, formados por signos, e que geram e interpretam signos. Esta perspectiva estende a proposta original de Souza [SOU 93] e apresenta uma justificativa mais poderosa para a Engenharia Semiótica, pois revela que a engenharia de software e as atividades de programação e de interação proporcionam diferentes interpretações da usabilidade de um software. Estas considerações justificam a necessidade de se conceber o sistema computacional através de uma abordagem semiótica baseada na produção de signos.

Entretanto, esta aplicação apenas ocorre quando o usuário tiver conhecimento sobre:

a) a funcionalidade do sistema – quais as funções relacionadas que o sistema deve executar;
b) o modelo de interação – quais ações podem ou devem ser feitas.

Estes modelos são abstratos e apenas podem ser construídos, comunicados e interpretados com o auxílio de sistemas de signos, o que pode ocorrer através de manuais, sistemas de ajuda ou da interface de usuário com a utilização dos signos de interface (pode-se ver mais detalhes na seção 2.5.2). Quando este processo ocorre através da interface de usuário durante o processo de utilização, podemos dizer que ela é a expressão de um signo cujo objeto é o modelo de usabilidade necessário para a aplicação do sistema.

O objeto é a aplicação do sistema na resolução de problemas do domínio. Dizemos, portanto, que quando o sistema, através da sua interface de usuário, gera no usuário um processo mental que o capacita a aplicá-lo na resolução de problemas ele é um *signo*. Esta perspectiva nos permite observar ***aplicações de software como signos***.

Podemos observar a importância da interface para que o usuário adquira o modelo de usabilidade de um sistema. Por este motivo, o *design* da aplicação requer a produção de interfaces que comuniquem o modelo de usabilidade. O requisito fundamental que a perspectiva da Engenharia Semiótica coloca para o desafio de usabilidade é que o *design* é uma atividade de produção de signo.

O que a Engenharia Semiótica diz é que no *design* do modelo de interação deve-se estar realizando uma comunicação. Por exemplo, um botão serve ao acionamento (interação) ao mesmo tempo que a sua imagem (*affordance*) comunica qual é a ação que ele deve fazer (metacomunicação). Ele é expressão e conteúdo.

A função metacomunicativa pode ser *direta*, quando o *designer* envia uma mensagem diretamente através da interface utilizando um signo cuja função seja exclusivamente esta, ou *indireta*, quando o *designer* envia suas mensagens através de objetos (*widgets* ou teclas) que têm primordialmente função de acionamento ou de revelação e que, nesta situação, adquirem a função de signo.

O *design* de interfaces de usuário é um processo comunicativo porque ele se utiliza da interface e do processo de interação que ocorre através dela como *medium* para a elaboração da expressão do signo. Para que o *designer* possa estabelecer a comunicação com o usuário, ele deve produzir distinções na interface que sejam a expressão da sua mensagem. Como a interface tem um potencial ilimitado para a produção de signos, é preciso que o *designer* disponha de tipo-signos para restringir os interpretantes do usuário "em torno" do Modelo de Usabilidade concebido.

Segundo Leite [LEI 98], a interface oferece um *medium* (contínuo expressivo) chamado de ***Medium Interface*** (MI), onde as expressões dos signos, como os componentes da Mensagem do *Designer*, podem ser construídos. O conceito de *Medium Interface* refere-se ao ambiente físico da interface por onde são veiculados mensagens ou signos da interface. É através dele que os signos são interpretados pelo usuário. O exemplo mais comum de *medium* é a tela onde são veiculados os signos produzidos pelos gráficos computacionais, porém qualquer dispositivo em que o usuário perceba distinções e as possa interpretar como signos pode ser considerado como *medium interface*.

2.5.2 Os Signos de Interface

Signo de Interface (SI), segundo Leite [LEI 98], são os signos que dizem respeito a qualquer distinção simbólica que adquira significado para o usuário ou para o *designer*. Ele representa qualquer elemento, articulado ou não, que pode ser veiculado no *medium interface* tais como *widgets*, ícones, palavras, teclas, LEDs, menus, caixas de diálogo, assistentes de tarefas, e vários outros, podendo ser, ainda, os resultados de uma computação, os comandos e dados digitados por usuários, ou os arrastos e seleções com o mouse, possibilitando, assim, comunicar ao usuário as funcionalidades do sistema e o modelo de interação.

Um SI é uma abstração teórica sobre entidades computacionais de hardware ou software que podem enviar mensagens para o usuário e/ou podem interpretar comandos ou ações dos usuários.

Os signos de interface podem ter função de *acionamento, revelação ou metacomunicação*. A função de acionamento ocorre quando o signo permite que o usuário realize alguma ação que causa uma mudança no sistema. A função de revelação é utilizada para expressar os estados do sistema. A função de metacomunicação ocorre quando o usuário se utiliza das outras duas para enviar a sua mensagem para o *designer*. Isto pode ser feito pela aparência e comportamento dos signos de interface.

Utilizando os tipos-SI de um sistema semiótico, o projetista deve escolher a expressão mais apropriada de maneira a ativar a cadeia de interpretantes (o significado pretendido) na mente do usuário, que correspondam a elementos da funcionalidade ou do modelo de interação.

Um tipo-SI possui um tipo expressivo e um tipo semântico que definem a sua expressão e o seu significado, respectivamente.

O tipo expressivo é definido através de suas propriedades visuais estáticas e dinâmicas. Estas propriedades definem o que é conhecido por *aparência e comportamento* de *widget* (*look and feel*). A aparência é determinada pelas propriedades visuais dos signos de interface. O comportamento é determinado pelo software da interface e oferece *feedback* para o usuário ou representações dinâmicas.

O tipo semântico determina como a expressão é associada aos elementos do Modelo de Usabilidade (as ações que ele permite fazer e o efeito que terá no sistema) e

mensagens de metacomunicação do *designer*. O que define a base do significado de um SI é um programa de computador que interpreta as ações de interface emitidas através de *ferramentas de acionamento* e realiza uma outra ação (seu efeito) que modifica o estado do sistema (da funcionalidade e/ou da própria interface).

3 Arquitetura do Ambiente Educacional

3.1 Modelo Pedagógico do Ambiente

O modelo pedagógico proposto nesta pesquisa segundo Andrade et al. [AND 2000] e [AND 2001] está fundamentado na busca por uma forma colaborativa de aprendizagem, conforme mostrado por Jaques et al. [JAQ 2002], que se efetive através da interação social. As interações podem ser de vários tipos, considerando critérios como a temporalidade, quanto ao número de participantes, quanto à reciprocidade, quanto à hierarquia e até mesmo perceptual baseada nas características comportamentais, personalidade, motivações e estado emocional dos indivíduos. Quanto à temporalidade, a interação pode ser síncrona ou assíncrona. Quanto ao número de participantes, a interação pode ser biunívoca do tipo "um para um", "um para todos" e "todos para todos". Quanto à reciprocidade, a interação pode ser recíproca ou não recíproca. Na interação hierárquica, é feita uma análise prévia dos participantes classificando-os entre superior, inferior e moderado. Além disso, a interação social pode ser motivada por vários fatores perceptuais, atitudes interpessoais e relação de papéis considerando elementos como idade e características de personalidade. As características perceptuais identificadas são o esforço, a confiança e a independência que influem diretamente na motivação, de acordo com Bercht et al. [BER 99]. Em Vicente et al. [VIC 98] podemos ver possíveis formas de se diagnosticar a motivação dos alunos em STI.

As estratégias de ensino que poderão ser utilizadas pelo sistema ainda estão em fase de estudo e decisão pelo grupo (por exemplo, ao aplicar exercícios, usa-se o modelo de ensino por treinamento). O mais importante a ser ressaltado é que se privilegie a interação social entre os integrantes da sociedade, independente das estratégias ou táticas que venham a ser utilizadas.

Conforme estudo apresentado por Bercht, para o processo de ensino-aprendizagem é necessário também um processo de avaliação, pois, a partir da avaliação, dependerão as ações pedagógicas seguintes (estratégicas e táticas) para melhorar a qualidade do processo de ensino-aprendizagem. Assim, pode-se citar alguns exemplos de avaliação que podem vir a ser utilizados pelo ambiente proposto: a avaliação na área afetiva (objetivos de atenção, objetivos evidenciadores da reação do aluno, entre outros); técnica da observação, para motivar e verificar as emoções dos alunos, de forma a atualizar o *profile* afetivo no modelo do aluno; a avaliação na área cognitiva, utilizando testes de desempenho como testes objetivos (completar lacunas, respostas curtas, questões de certo ou errado, questões de associação, questões de ordenação, ou ainda questões de múltipla escolha), aumentando ou diminuindo o nível de dificuldade. Maiores detalhes sobre avaliações podem ser vistos no estudo feito por Bercht em [BER 97].

3.2 Modelo Computacional do Ambiente

O sistema proposto inicialmente em Andrade et al. [AND 2000] e [AND 2001] era formado por quatro classes de agentes artificiais – o agente ZDP, o agente mediador, o agente social e o agente semiótico – e os agentes humanos (aprendizes). O sistema atual sofreu evoluções, e hoje em dia, ele é composto por agentes humanos (aprendizes

e tutores) e por cinco classes de agentes artificiais: o agente diagnóstico, o agente mediador, o agente colaborativo, o agente social e o agente semiótico, todos fazendo parte de uma sociedade de agentes conforme explicado na seção 2.4.4.

Os agentes artificiais monitoram e auxiliam os agentes humanos em suas atividades colaborativas. No sistema proposto, todos os personagens, aprendizes e agentes, são modelados como agentes sociais, conforme comentado na seção 2.4.1, integrados em um ambiente de aprendizagem colaborativa. Na FIGURA 3.1, é ilustrada a arquitetura do sistema proposto.

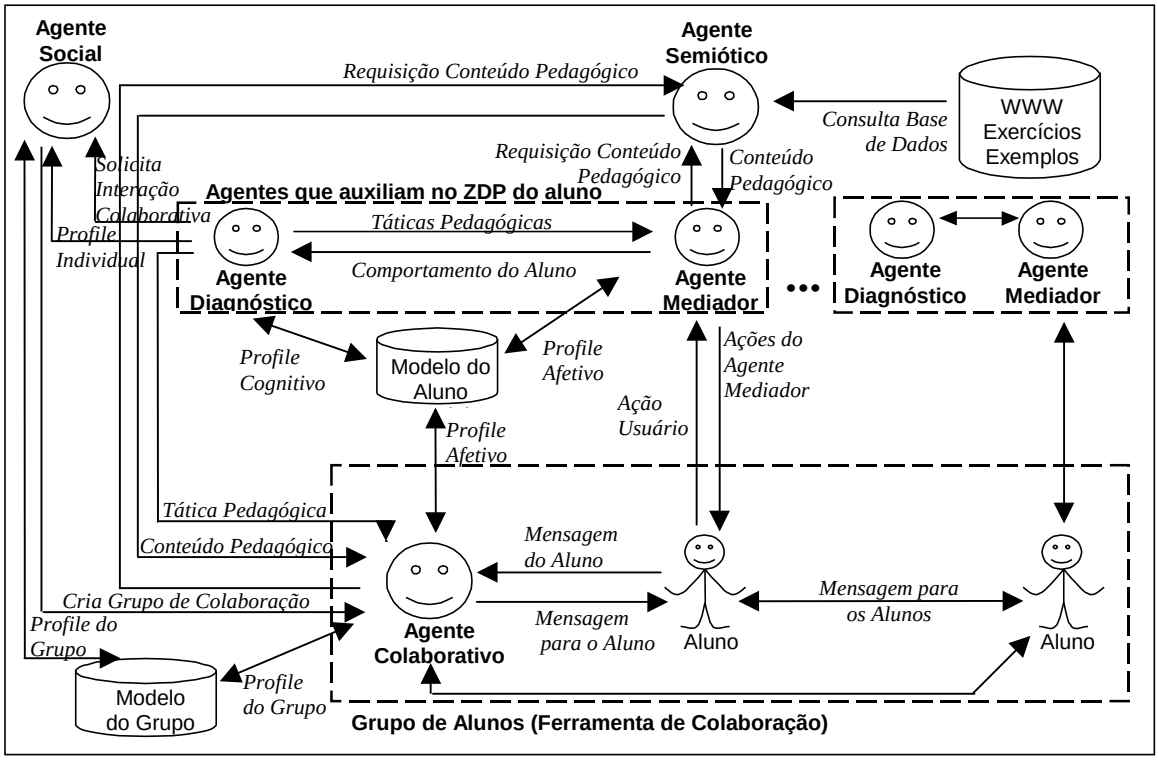

FIGURA 3.1 – Sociedade de Agentes de Colaboração num Ambiente de Aprendizagem

Pode-se observar na figura, que os agentes diagnóstico, mediador e colaborativo são responsáveis por monitorar e interagir com os aprendizes. Os agentes semiótico e social auxiliam nas atividades relacionadas à toda a sociedade. Assim, existe um agente diagnóstico e um agente mediador para cada aluno, um agente semiótico e social para toda a sociedade, e um agente colaborativo para cada grupo de alunos formado com características em comum.

Embora os agentes diagnóstico e mediador interajam entre si, apenas o agente diagnóstico interage com o agente social em busca de parceiros para auxiliar um determinado aluno em sua aprendizagem. Os agentes mediadores irão interagir com o agente semiótico para obter os signos e símbolos que serão apresentados aos alunos. Neste processo de identificação da necessidade de apresentar novos símbolos, apenas o agente diagnóstico pode modificar o modelo do aluno no que refere os aspectos cognitivos para atualização deste (*profile* cognitivo), enquanto que os agentes mediadores e colaborativos atualizam o modelo do aluno em termos de aspectos afetivos (*profile* afetivo).

A interação do sistema com o aprendiz é realizada através do agente mediador. Este verifica a ação do estudante, repassando o comportamento do estudante para o agente diagnóstico, que vai verificar a capacidade de aprendizagem do aluno, conforme

as informações do modelo do aluno, atualizando o modelo (*profile* cognitivo), se for preciso. O agente diagnóstico, baseado nas informações do modelo do aluno e nas habilidades de aprendizado, pode verificar uma deficiência do aprendizado do aluno que seja interessante a atividade em grupo. Assim, o agente diagnóstico envia informações de perfil individual (*profile* individual) para o agente social de forma que o agente social possa identificar outros alunos com mesmas afinidades (por exemplo, conhecimento e crenças) e dificuldades comuns, formando um mesmo grupo de trabalho. Assim, é criado um grupo de colaboração através de um agente colaborativo. Estes alunos irão se reunir através de uma ferramenta de comunicação síncrona para troca de idéias e conhecimentos. O agente colaborativo irá monitorar e mediar a interação entre os alunos, corrigindo concepções errôneas, sugerindo novos conteúdos e incentivando a participação. Este conhecimento que o agente colaborativo adquire do grupo é atualizado no modelo de grupo.

Para o sistema, qualquer participante humano, representado na FIGURA 3.1 apenas pelo aluno, pode assumir o papel de tutor ou aluno.

O sistema também pode funcionar como um tutor inteligente, onde o agente diagnóstico envia uma nova tática de ensino para o agente mediador, que por sua vez, vai solicitar ao agente semiótico que faça uma busca na base de dados de novos signos e instrumentos capazes de auxiliar no aprendizado cognitivo do aluno. O agente semiótico após escolher os signos que melhor se enquadram à solicitação do material instrucional, vai montar uma página HTML (*HyperText Markup Language*) dinamicamente, repassando-a ao agente mediador. Este por sua vez, pode incluir artifícios de motivação, exibindo assim o conteúdo final para o estudante.

Dentre as várias formas de interação envolvidas no modelo, pode-se enumerar: interação entre agentes artificiais; interação entre agente artificial e agente humano; e interação entre agentes humanos (aprendizes e tutores).

Convém ressaltar, ainda, que nesta arquitetura, tanto o agente mediador, quanto o agente colaborativo serão implementados como agentes animados, conforme estudo feito por Jaques et al. [JAQ 2001].

Em termos de arquitetura computacional, pode-se dizer, conforme explicado na seção 2.4.3, que temos o seguinte: o agente semiótico é um agente com características reativas e cognitivas, possuindo arquitetura híbrida, pois reage aos eventos que ocorrem no ambiente e possui uma capacidade de raciocínio “fraca” (em termos de tomar decisão de quais signos deve apresentar), porém não toma iniciativa sozinho; o agente diagnóstico possui arquitetura deliberativa e arquitetura BDI (SEM), e os demais agentes possuem arquitetura deliberativa, desenvolvendo planos e tomando decisões; e dessa forma, da arquitetura individual dos agentes emerge uma arquitetura global do projeto com características híbridas, sendo o projeto como um todo uma arquitetura híbrida.

A comunicação entre os agentes, conforme visto na seção 2.4.5, ocorre na grande maioria, por passagem de mensagem ponto-a-ponto, exceção feita para o agente social que, pelo fato de ter que se comunicar com todos os agentes diagnósticos, a fim de encontrar alunos com características semelhantes, pode passar a mensagem para grupos de agentes (passagem de mensagem *multicast*).

3.3 Agente Diagnóstico

O agente diagnóstico visa implementar, o mais próximo possível, o conceito de Zona de Desenvolvimento Proximal defendida por Vygotsky, visando transformar habilidades potenciais em habilidades reais e buscando expandir a capacidade de desenvolvimento sócio-cognitiva do aluno conforme foi explicado na seção 2.3.

Assim, o sistema possui agentes diagnóstico responsáveis por observar o desenvolvimento real dos aprendizes e propor atividades que tornem as suas capacidades reais (Nível de Desenvolvimento Real) o mais próximo possível dos níveis desejados (Nível de Desenvolvimento Potencial). Este processo pode ser melhor visualizado na FIGURA 3.2:

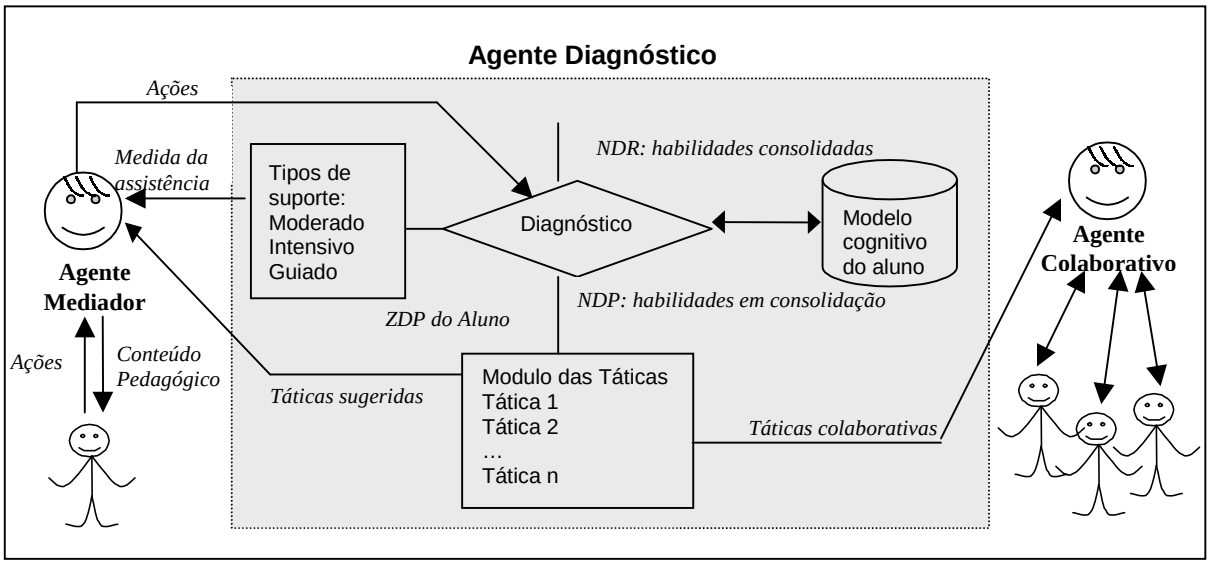

FIGURA 3.2 – Arquitetura Interna do Agente Diagnóstico

Pode-se observar que as ações realizadas pelo usuário são repassadas do agente mediador para o agente diagnóstico. Este realiza um diagnóstico levando em consideração as habilidades consolidadas e as habilidades em consolidação (NDR e NDP), utilizando-se do modelo cognitivo do aluno, e conseqüentemente sugerindo determinado tipo de suporte e táticas a serem utilizados pelo agente mediador, ou ainda, pode repassar táticas colaborativas a serem usadas pelo agente colaborativo.

Como cada agente deve monitorar especificamente um aprendiz, haverá para cada aprendiz um agente diagnóstico.

O agente diagnóstico, inspirado na teoria de Vygotsky [VYG 98] e [VYG 98a], e Baquero [BAQ 98], é responsável por estimular aquelas funções do aprendiz que ainda não amadureceram, mas que estão em processo de desenvolvimento. O agente diagnóstico poderá ter como funções: variar o grau de controle das atividades conjuntas, propor tarefas progressivamente, ou modificar as formas de ajuda/suporte oferecidos, definindo as táticas a serem utilizadas.

Para auxiliar no processo de aprendizagem, o agente diagnóstico deve possuir um modelo do aluno, identificando suas habilidades e deficiências, que será construído pela observação das ações do usuário. Desta maneira, ele é capaz de indicar a extensão na qual o aluno pode expandir-se, acessando e modificando este modelo.

O modelo do aluno contém as seguintes informações: traço de personalidade, baseado nos trabalhos de Castelfranchi et al. [CAS 97], e Bercht et al. [BER 99], esforço, confiança, independência do aluno. É importante ressaltar que as crenças dos alunos são deduzidas das interações e que, em um grupo colaborativo, crenças individuais dos alunos incluem também crenças em relação aos outros membros do grupo. Para tanto, é necessário um formalismo que seja capaz de fazer inferências e representar as informações dos alunos como traços de personalidade e que permita representar crenças como, por exemplo, o modelo X-BDI de Mora et al. [MOR 98], ou ainda os trabalhos feitos por Bordini et al. [BOR 2001] com a utilização de *Design-To-Criteria* (DTC) onde permite que planos que sejam candidatos de seleção de intenção sejam melhor escolhidos para serem executados em agentes com arquitetura BDI.

O agente diagnóstico encontra-se em desenvolvimento como tema da tese de doutorado de Adja Ferreira de Andrade.

3.4 Agente Mediador

O agente mediador é responsável por realizar a interface entre o sistema e o aprendiz. Ele auxilia no processo de internalização do aluno decorrente do contato com o ambiente social de EAD.

A diferença central entre o agente mediador e o agente diagnóstico é que o primeiro realiza todas as tarefas de interface e comunicação com o usuário. O agente diagnóstico, por sua vez, é responsável pelo processo de aprendizagem, ou seja, a construção do modelo do aluno (*profile* cognitivo), e pela identificação de deficiências na aprendizagem através da *observação* das ações do usuário.

O agente mediador será um agente pedagógico animado (conforme visto na seção 2.4.2), onde além da função de mediar a interação do aprendiz com o agente diagnóstico, deverá acessar e atualizar o modelo do aluno (*profile* afetivo), auxiliando na predição de comportamentos deste usuário, através de incentivo, motivação, percepção e exibição de emoções, a fim de mediar as melhores ações a serem executadas para auxiliar no processo de aprendizagem do aluno.

Este agente pedagógico animado encontra-se em desenvolvimento como tema da dissertação de mestrado de Everton Weber Bocca.

3.5 Agente Colaborativo

O agente colaborativo tem por função promover e mediar a interação entre grupos de alunos em ferramentas síncronas de comunicação como por exemplo, o *chat*. Para isso, ele assiste os alunos durante as interações, incentivando-os quando se mostrarem desmotivados, apresentando novos conceitos e corrigindo concepções errôneas.

Devido a sua função social, no que diz respeito a comunicar-se com o usuário e promover e monitorar a interação entre alunos, é interessante que esse agente possua uma interface onde seja possível explorar a natureza social do homem. Assim, ele será representado como um agente animado possuindo identidade e interagindo com o aluno através de linguagem natural, mais especificamente reconhecimento de padrões. Uma vez reconhecido determinado padrão, será repassado ao agente semiótico a fim de verificar se o mesmo faz parte do conteúdo pedagógico do aprendizado.

Da mesma forma como nas interações sociais humanas, o agente colaborativo deve perceber e exibir emoções, promovendo assim o desenvolvimento emocional e

afetivo do aluno, gerando-lhe autoconfiança e um estado de espírito positivo, oportunizando uma melhor aprendizagem. Para isso, é necessário o modelo do aluno possuir características emocionais como o modelo definido por Bercht et al. [BER 99].

Isto vem ao encontro da afirmação de Minsky [MIN 85] onde cita o seguinte:

"A questão não é se máquinas inteligentes podem ter alguma emoção, mas se máquinas podem ser inteligentes sem nenhuma emoção".

Estudos sobre motivação e emoções existentes em interações sociais podem ser vistas em Canamero & Velde [CAN 97].

A FIGURA 3.3 mostra a arquitetura interna do agente colaborativo.

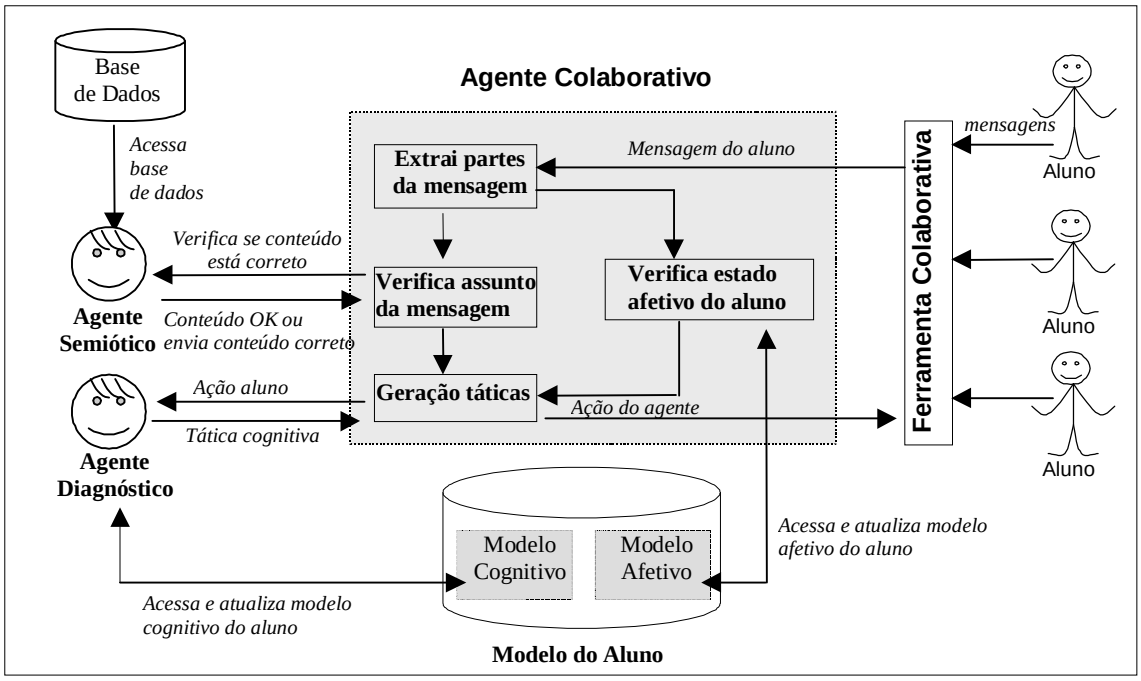

FIGURA 3.3 – Arquitetura Interna do Agente Colaborativo

Nesta arquitetura, pode-se observar que, a partir das mensagens trocadas entre os alunos na ferramenta colaborativa, o agente colaborativo extrai parte destas mensagens verificando o assunto contido nas mesmas. A partir disso, o agente colaborativo troca mensagens com o agente semiótico a fim de verificar se o conteúdo tratado está correto. O agente colaborativo, então, verifica o estado afetivo do aluno consultando e atualizando o modelo afetivo do aluno. Por fim, utilizando-se da tática cognitiva sugerida pelo agente diagnóstico, o agente colaborativo gera uma ação a ser realizada na ferramenta colaborativa.

O agente colaborativo encontra-se em desenvolvimento como tema da tese de doutorado de Patrícia Augustin Jaques.

3.6 Agente Social

O agente social conhece todos os agentes diagnóstico da sociedade, como também estes têm conhecimento da sua existência. Sua função é estabelecer a integração da sociedade e construir o modelos de grupos de aprendizes. Uma das atividades do agente social é investigar a existência de um outro aprendiz que tenha além do conhecimento necessário, crenças e traços de personalidade que propiciem a melhor cooperação entre os aprendizes.

Na aprendizagem colaborativa, o grupo é uma entidade ativa e, por isso, o sistema deve conter informações que o identifiquem como um todo. Estas informações compõem o modelo do grupo e são coletadas e armazenadas pelo agente social. Um modelo de grupo pode ser construído a partir dos modelos individuais dos alunos obtidos através da interação com os agentes diagnóstico, como também das observações dos grupos pelo agente colaborativo. Um modelo de grupo é composto pelas seguintes informações: concordância, afetividade, crenças do grupo, autoconfiança e conflito.

Maiores detalhes sobre utilização de modelo de grupo num ambiente de aprendizado colaborativo pode ser visto em Paiva [PAI 97], e Barros & Verdejo [BAR 99].

3.7 Agente Semiótico

O ***agente semiótico,*** que é o objetivo desta pesquisa, é responsável pela utilização de símbolos (signos e instrumentos), conceitos e linguagem que serão repassados como material instrucional para o aluno.

Para que o agente mediador possa cumprir o seu papel, é necessária a intervenção de estímulos externos na forma de instrumentos e signos. Estes elementos são introduzidos pelo agente semiótico para auxiliar na atividade cognitiva do aluno a fim de solucionar um dado problema.

O agente semiótico faz uso dos signos e instrumentos para auxiliar a atividade cognitiva do aluno. Para tanto, o agente usa vários signos, dentre eles: o desenho, a escrita, o sistema de números, figuras, recursos multimídia, hipertextos e animações.

Devido a importância deste agente, ele está melhor detalhado no capítulo 4.

3.8 Agente Humano

Os agentes humanos são visualizados como agentes que estabelecem relações sociais entre si de acordo com as suas características pessoais e personalidade. Desta maneira, é importante que traços e características de personalidade do aprendiz estejam contidos em seu modelo, conforme Castelfranchi et al. [CAS 97], já que esses traços irão afetar diretamente a interação através dos papéis que cada aprendiz irá exercer. Estes papéis permitem determinar relações de afetividade que irão acontecer em um grupo de alunos levando, por exemplo, aprendizes a formarem sempre o mesmo grupo de trabalho baseado nas afinidades (por exemplo, conhecimento e crenças) e dificuldades comuns. O conhecimento da personalidade é necessário para entender as ações realizadas pelos agentes e as relações entre os agentes durante a interação.

3.9 Interação entre os Agentes

A interação entre os agentes não está restrita ao modelo computacional proposto. Ao contrário, a interação computacional entre os agentes artificiais visa contribuir ainda mais para a comunicação e a troca entre os agentes humanos. A interação social será um dos objetivos principais deste modelo, visto a proposta tratar-se de um modelo de aprendizagem colaborativa. Dentre as várias formas de interação envolvidas no modelo podemos enumerar: interação entre agentes artificiais; interação entre agente artificial e agente humano e interação entre agentes humanos (aprendizes e tutores).

Inicialmente, o agente diagnóstico está diretamente relacionado com o agente social e com o fato de que a interação com este agente pode favorecer o desenvolvimento intelectual do aluno a partir da interação com o ambiente.

Quando for observado que o aprendiz não está tendo o nível de aprendizagem desejado, o sistema funciona como um tutor inteligente. O agente diagnóstico, neste momento, notificará ao agente mediador que fará a solicitação ao agente semiótico. O agente semiótico deve escolher na base de dados, o signo que melhor se enquadre àquela determinada situação e repassá-lo ao agente mediador que deve exibi-lo ao usuário. Dessa forma, o agente mediador estará presente dentro do esquema estímulo - elo de mediação – resposta.

No aspecto de comunicação entre os agentes humanos, o sistema disponibiliza ferramentas (síncronas ou assíncronas) quando a presença física não for possível (por exemplo, no caso de aulas virtuais). É importante ressaltar que os agentes deste sistema não realizam nenhuma tarefa de mediar a interação entre os agentes do ambiente, através de tarefas de negociação, argumentação ou resolução de conflitos. Maiores detalhes sobre isto podem ser vistos em Sycara [SYC 98]. Para um maior aprofundamento sobre conflitos meta-cognitivos que ocorrem no modelo de grupo, pode ser consultado o trabalho de Tedesco [TED 98]. Maiores detalhes em termos de coalizões em simulações sociais baseadas em agentes pode ser visto nos trabalhos de David et al. [DAV 99] e [DAV 2000]. Estudos sobre reconhecimento de potencial de cooperação, formação de coalizão, formação de planos e ação de coalizão podem ser melhor detalhados em Sichman & Conte [SIC 98a]. Maiores detalhes em termos de estudos sobre negociação entre agentes envolvendo idéias de concordabilidade e harmonia podem ser vistos em Oliveira [OLI 95]. Um estudo sobre decomposição do problema (decomposição extrínsica e intrínsica) pode ser visto em Alvares et al. [AMD 99].

4 Agente Semiótico

Este capítulo inicia com a justificativa para chamar o agente desenvolvido neste trabalho de Agente Semiótico. Este agente é o responsável por escolher os signos (conforme visto nas seções 2.5.1 e 2.5.2), a partir da base de dados, que melhor representarão a função de metacomunicação (conforme visto na seção 2.5) do conteúdo de ensino-aprendizagem a ser mostrado para o aluno. A semiótica é a ciência que estuda os signos, sistemas de signos, significação, comunicação e todos os processos culturais, segundo Eco [ECO 80], conforme apresentado na seção 2.5. Além disto, o sistema, como um todo, inspira-se na teoria sócio-interacionista de Vygotsky, cuja teoria utiliza-se de signos como processo mnemônico para o aprendizado através da interação social, conforme apresentado na seção 2.3.

Assim, este agente é chamado de Agente Semiótico pois é o responsável pela busca dos signos necessários para o processo de ensino-aprendizagem do aluno, ou seja, o agente responsável em enviar as informações e mensagens (material instrucional) para o usuário (aluno).

O agente semiótico é um agente que está inserido numa sociedade de agentes possuindo as seguintes propriedades, conforme visto anteriormente na seção 2.4.2:

- é um agente **autônomo**, pois consegue agir na sociedade por meios próprios e controlando suas próprias ações;
- possui **habilidade social** interagindo com outros agentes, como o agente mediador e agente colaborativo;
- é **reativo**, pois reage a estímulos de solicitação de conteúdo do agente mediador e agente colaborativo;
- é **contínuo**, pois consegue permanecer continuamente na sociedade;
- é **comunicativo**, pois troca mensagens com outros agentes (agente mediador e agente colaborativo);
- é **racional**, embora seja uma racionalidade “fraca”, baseado em regras de decisão, pois possui a capacidade de tomar decisões, em relação a quais signos, ou seqüência de signos, é melhor para ser apresentado na atividade cognitiva do aluno;
- é **flexível**, pois permite a intervenção de outros agentes (agente mediador e agente colaborativo).

O agente semiótico é um agente que possui arquitetura híbrida (conforme visto na seção 2.4.3), possuindo um pouco da arquitetura deliberativa em termos de deliberação sobre diferentes opções, flexibilidade, comunicação, raciocínio, tomando decisões sobre quais signos, dentre todos existentes na base de dados, que deve ser apresentado ao aluno em um dado momento (embora seja um agente cognitivo “fraco”, pois possui tomada de decisões, mas não possui aprendizado); e também possui um pouco da arquitetura não deliberativa em termos de reação a estímulos dados pelos outros agentes, ou seja, conforme a solicitação dos outros agentes, através das mensagens de comunicação, de que deve ser apresentado um determinado conteúdo, é que o agente semiótico vai começar a agir.

Para que o agente mediador possa cumprir o seu papel junto ao aprendiz, é necessária a intervenção de estímulos externos na forma de instrumentos e signos. Estes elementos são introduzidos pelo agente semiótico para auxiliar na atividade cognitiva do aluno a fim de solucionar um dado problema.

O agente semiótico faz uso dos signos e instrumentos, a partir da base de dados e da solicitação da intervenção feita pelo agente mediador, para auxiliar a atividade cognitiva do aluno. Para tanto, o agente usa vários signos, expressos das mais diversas formas, dentre eles: desenho, escrita (apresentando o domínio em forma de parágrafos, exemplos, citações, tabelas, palavras-chave, exercícios), sistema de números, figuras, recursos multimídia e animações em forma de *links* disponíveis, páginas HTML, com ou sem hipertexto, *links* de outras páginas sobre um dado conteúdo, propiciando, assim, a apresentação do material instrucional conforme a tática de ensino especificada pelo agente diagnóstico.

O agente semiótico é o responsável pela interface de alimentação da base de dados, verificação da existência do *link* para futura utilização, possuindo regras para a tomada de decisão sobre quais signos melhor representam a função de comunicar o conteúdo a ser ensinado ao aprendiz.

O agente semiótico constrói dinamicamente a página a ser apresentada ao aluno, como pode ser visto no capítulo 5, mostrando conteúdos mais específicos à medida que o aluno vai se aprofundando no detalhamento do assunto a ser ensinado.

Diferentemente dos outros ambientes de ensino-aprendizagem, este caracteriza-se por apresentar uma forte abordagem de comunicação entre os agentes, conforme visto na seção 2.4.5: ao invés da maioria das informações ficarem armazenadas na base de dados, em que todos ou grande parte dos agentes pode pesquisar as informações necessárias, a troca de informações através da comunicação entre os agentes tornou-se um fator de primordial importância para o funcionamento do sistema. Exemplos de mensagens trocadas entre os agentes podem ser vistos na seção 5.1.

Embora as atividades sociais sejam de responsabilidade de outros agentes (agente social e agente colaborativo), o agente semiótico disponibiliza instrumentos que viabilizarão a interação social entre alunos e professores:

- e-mails de alunos e professores;
- *chat* incluindo um *log* das informações tratadas;
- agendamento de reuniões;
- FAQ (perguntas mais freqüentes);
- informações de outros alunos/professores que utilizam ou utilizaram o sistema.

4.1 Arquitetura Interna do Agente Semiótico

A FIGURA 4.1 mostra a arquitetura interna do agente semiótico. Pode-se observar, nesta arquitetura, que o agente semiótico, a partir da solicitação de conteúdo pedagógico vindo do agente colaborativo ou agente mediador, verifica quais são as táticas, preferências e nível do aluno, procurando na base de dados quais são os signos ideais a serem utilizados para o conteúdo pedagógico.

Baseado nestas informações, o agente semiótico constrói dinamicamente o conteúdo aplicando um estilo ideal (*style sheet*) sobre o arquivo XML (*eXtensible Markup Language*), gerando assim o material instrucional com características personalizadas em forma de página HTML como resposta para o agente mediador.

Além disto, o agente semiótico pode enviar uma mensagem para o agente colaborativo em forma de mensagem KQML dizendo se o padrão reconhecido pelo agente colaborativo, durante as trocas de mensagens entre os alunos, faz parte de determinado conteúdo a ser tratado no processo de ensino-aprendizado.

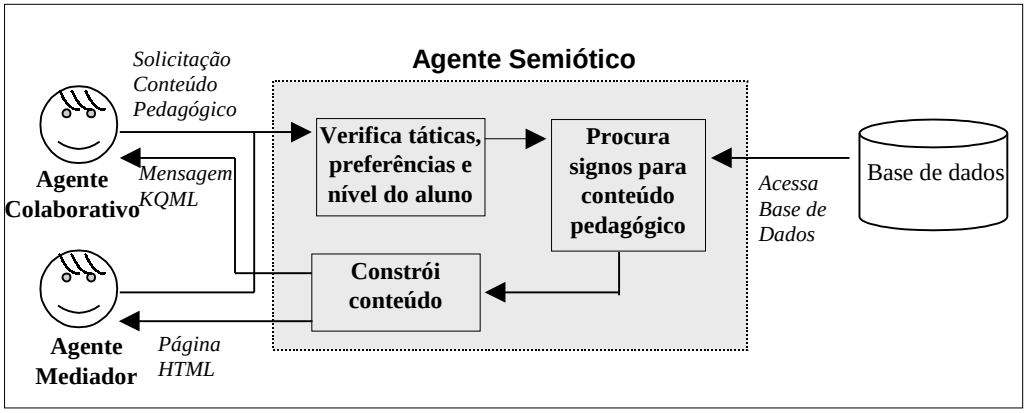

FIGURA 4.1 – Arquitetura Interna do Agente Semiótico

4.2 Agente Semiótico e Engenharia Semiótica

Tomando como base a abordagem da seção 2.5, considera-se de agora em diante o termo *designer* como sendo o ***agente semiótico*** com o papel de *designer*, ou seja, o agente semiótico desempenhando o papel de projetista de interface. O agente semiótico decide quais os signos que melhor se enquadram, dado uma determinada situação, ou seja, dependendo da tática de ensino especificada pelo agente diagnóstico e repassada pelo agente mediador.

Para isso, é importante ter-se uma modelagem de quais signos serão utilizados e como apresentá-los ao usuário. Assim, baseou-se esta modelagem no trabalho proposto por Leite [LEI 98], e Leite & Souza [LEI 99], para a especificação das regras de decisão do agente semiótico, que são utilizadas por ele na construção do material instrucional e da interface de usuário. Para tanto, é utilizado a Linguagem de Especificação da Mensagem do *Designer* (LEMD), cujo objetivo é apoiar a formulação da mensagem sobre o modelo de usabilidade.

A LEMD permite especificar os *signos do domínio, as funções da aplicação e os comandos de função* como sendo mensagens de metacomunicação enviadas pelo agente semiótico na construção do material instrucional e da interface de usuário. Além disso, o agente semiótico pode utilizar diversas outras mensagens que comuniquem ao usuário o conjunto de tarefas que ele tem condições de realizar, o processo de navegação entre as páginas (telas) da interface, e informações diretas que ajudem ao usuário interagir com o sistema. Esta estrutura básica permite diferenciar entre os signos de interface que são essenciais para usuário utilizar o sistema e as mensagens que organizam estes signos ou que auxiliam o usuário no aprendizado cognitivo.

O agente semiótico baseia-se na LEMD para mostrar diversos tipos de mensagens que foram propostas por Leite [LEI 98] para um *designer* humano[5]:

- mensagens sobre estados de signos do domínio - revelam o estado do sistema e permitem ao usuário avaliar se a sua meta foi atingida;
- mensagens sobre funções da aplicação - revelam o seu estado operacional e o que o usuário deve fazer para controlá-la;

[5] Um exemplo mais aprofundado pode ser visto em Leite [LEI 98] e Leite & Souza [LEI 99].

- mensagens sobre a estrutura sintática dos comandos - revelam a estrutura sintática (interações básicas) e a articulação das interações (seqüência, repetição, agrupamento, combinação e seleção) que o usuário precisa desempenhar;
- mensagens sobre interações básicas - indicam ao usuário a interação (acionar, fornecer informação e selecionar informação) a ser desempenhada;
- mensagens de metacomunicação de assistência a tarefas - auxiliam ao usuário a realizar tarefas compostas por mais de um comando;
- mensagens de metacomunicação para apresentação e controle da leitura da mensagem - comunicam como o usuário deve "ler" a própria mensagem;
- mensagens de metacomunicação direta - permitem enviar uma mensagem diretamente ao usuário para se referir a qualquer outro elemento da interface, inclusive à própria mensagem.

Além disso, utilizou-se as idéias do trabalho de Martins & Souza [MAR 98] para especificar uma linguagem visual, por exemplo, utilizar cores para expressar as propriedades de objetos. Isto permite estruturar o meio de expressão com características mínimas de código, correlacionando elementos da expressão e do conteúdo de forma sistemática. Na medida em que esta correlação incorpora-se num certo padrão cognitivo, o signo visual do agente semiótico tem melhores possibilidades de ser reconhecido e o usuário possui maior facilidade em expressar suas intenções, atingindo mais facilmente os objetivos de usabilidade do sistema. Isto é facilmente implementado com o uso de XML e *style sheets* no momento da criação dinâmica das páginas de conteúdo do usuário.

A seguir, serão vistos exemplos de utilização da LEMD para o agente semiótico, como apresentado por Jung et al. [JUN 2001].

```
Command-Message Exercício for Application-Function Exercícios
 Join { Sequence { View Information-of Enunciado do exercício
                   Repeat { View Information-of Itens/Opções do exercício }
                   Join { Enter Information-of Resposta do aluno
                          Activate Show Command-Message Resolução/gabarito}}
        Select { Activate Start Application-Function Exercícios
                 Activate Waive Application-Function Exercícios } }
```

Neste caso, temos o agente semiótico responsável em apresentar um exercício para o usuário. Na realidade, um exercício é formado pela junção (*join*) das seguintes opções (ou seja, as informações não precisam ser fornecidas numa ordem específica, isto é, a opção "a" pode ser escolhida antes da "b" ou vice-versa):

a) uma seqüência de:
   - a apresentação do enunciado do exercício;
   - a repetição de apresentação (quantas vezes for necessário) dos itens do exercício;
   - uma junção (não é necessário uma ordem específica no fornecimento da informação) de:
     - entrar com a informação da resposta; e/ou
     - visualizar a resolução/resposta do exercício;

     Em outras palavras, o usuário pode entrar com a informação da resposta, e pode querer ver/aprender a resolução do exercício.
b) selecionar uma das opções:
   - ação de OK;
   - ação de Cancelar.

A seguir, é mostrado outro exemplo em que é apresentado o conteúdo instrucional ao aluno.

```
Command-Message Conteúdo for Application-Function Conteúdo_Pedagógico
  Join { Sequence { Repeat { Join { View Information-of Capítulo
                                    View Information-of Seção
                                    View Information-of Parágrafo
                                    View Information-of Html
                                    View Information-of Figura
                                    View Information-of Tabela
                                    View Information-of Lista
                                    View Information-of Exemplo
                                    View Information-of Citação
                                    View Information-of Link
                                    View Information-of Palavras-Chave
                                    Activate Show Command_Message Exercício}}
                     Repeat { View Information-of Bibliografia } }
           Select { Activate Previous Application-Function Conteúdo_Pedagógico
                    Activate Next Application-Function Conteúdo_Pedagógico } }
```

Explicando melhor o exemplo acima, pode-se dizer que mostrar um determinado conteúdo requer a seguinte seqüência de signos - é formado por uma junção (não importando a ordem) de:

a) obrigatoriamente uma seqüência de
   - uma repetição de
     - apresentação, não importando a ordem (*join*), das seguintes informações:
       - informação do Capítulo
       - informação da Seção
       - informação do Parágrafo
       - informação do HTML
       - informação de Figura
       - informação de Tabela
       - informação de Lista
       - informação de Exemplo
       - informação de Citação
       - informação de Link
       - informação de Palavras-Chave
       - mostrar um exercício (como visto no exemplo anterior)
   - uma repetição da informação de Bibliografia

b) selecionar uma das opções abaixo:
   - conteúdo Anterior;
   - Próximo conteúdo.

O resultado deste exemplo de utilização da LEMD para a geração dos signos necessários na apresentação do conteúdo instrucional ao aluno, pode ser visto, mais adiante, na FIGURA 5.11.

Assim, dependendo da mensagem recebida de outro agente como será mostrado na seção 5.1, e baseando-se nas ações a serem tomadas, conforme as regras de comportamento a serem definidas na seção 5.2, o agente semiótico, com o papel de *designer* na metacomunicação do material instrucional e da interface de usuário, vai utilizar-se dos exemplos mostrados acima para gerar os signos necessários para o processo de ensino-aprendizagem do aluno.

5 Implementação

Neste capítulo, tratamos da implementação do agente semiótico, apresentando as trocas de mensagens entre os agentes, as regras de comportamento do agente semiótico, a base de dados utilizada pelo sistema, o ambiente de gerência do material instrucional e o ambiente de educação a distância.

O agente semiótico está implementado em Java, [BIG 98], [DEI 2001], [SUN 98], [JAQ 2000], com o uso de *servlets*, onde um cliente envia uma solicitação de HTTP (*HyperText Transfer Protocol*) para o servidor (neste caso o agente mediador) que a recebe e envia para ser processada por *servlets* adequados (agente semiótico ou agente diagnóstico). Os *servlets* fazem seu processamento e a seguir retornam seus resultados para o cliente (na forma de documentos HTML).

Para armazenar todos os signos necessários para a geração do conteúdo pedagógico como o desenho, a escrita, o sistema de números, recursos multimídia, hipertextos e animações, a fim de apresentar os conteúdos em forma de capítulos, seções, parágrafos, páginas HTML, figuras, tabelas, listas, exemplos, citações, exercícios, entre outros, pensou-se, inicialmente, em utilizar XML gerando HTML, de forma a poder mostrar o mesmo conteúdo de maneiras diferentes, de acordo com as preferências dos alunos, ou ainda do nível em que se encontra o aluno, através da utilização de *style sheets* (folhas de estilo) que são, por exemplo, arquivos com extensão XSL (*eXtensible Style Sheets*) aplicado em arquivos XML. Detalhes sobre XML e XSL podem ser encontrados em Pimentel et al. [PIM 2000].

O modelo XML pode ser visto no Anexo 1, onde, a partir dele, aplicando-se o arquivo XSL, tem-se o resultado em HTML do material instrucional gerado pelo agente semiótico.

Porém, chegou-se à conclusão de que armazenar todo o conteúdo em um arquivo texto com extensão XML poderia vir a ser uma espécie de “gargalo” para o sistema, uma vez que, para se manutenir este arquivo “texto”, não seria muito viável. Por exemplo, para atualizar um determinado conteúdo (capítulo, seção, parágrafo, etc.) deletando-o ou inserindo novas informações, isto poderia levar muito tempo, ou seja, para se percorrer linhas seqüenciais num arquivo é muito mais demorado do que se as mesmas informações estivessem armazenadas em um banco de dados onde existem índices que permitem um acesso muitíssimo mais rápido, prático e eficiente.

Assim, foi construída uma base de dados que pode ser melhor visualizada na seção 5.3. Esta base de dados pode ser armazenada num banco de dados Oracle. Foi construído então um ambiente, em Java, que permite gerenciar todo o material instrucional, ou seja os signos, para o ensino-aprendizagem do aluno como pode ser visto na seção 5.4.

Para aproveitar a facilidade que oferece o XML e XSL para personalizar os dados de saída, o próprio ambiente de gerência do material instrucional gera, ao sair do sistema, um arquivo no formato XML correspondente a cada matéria.

Uma vez gerado este arquivo XML, é possível aplicar estilos de apresentação (*style sheets*), através do XSL, para formatar a saída, mostrando assim, por exemplo, os mesmos signos de forma diferente, dependendo do nível e preferência dos alunos em questão.

A partir deste arquivo XML, o agente semiótico gera o conteúdo instrucional (signos), baseado nos exemplos dados na seção 4.2, conforme as regras de

comportamento definidas na seção 5.2, transformando o resultado em um formato HTML como sendo o conteúdo pedagógico a ser mostrado para o aluno, ou, ainda, se for uma solicitação do agente colaborativo, o agente semiótico consultará a base de dados para verificar se o padrão encontrado nas mensagens trocadas entre os alunos faz parte do conteúdo em questão.

Os agentes trocarão mensagens entre si através de KQML utilizando RMI, conforme visto na seção 2.4.5. Exemplos destas mensagens são mostradas na seção a seguir.

5.1 Mensagens entre os agentes

Alguns tipos de conteúdo das mensagens (classe ContentMessage mostrado na seção 2.4.5) que serão trocadas entre os agentes podem ser vistos nas tabelas abaixo, com os seus respectivos parâmetros. Estas mensagens podem ser visualizadas na FIGURA 3.1 mostrada na seção 3.2.

TABELA 5.1- Mensagens do Agente Diagnóstico

Conteúdo de Entrada	**Conteúdo de Saída**
(Mensagem Comportamento do Aluno) Recebe o comportamento do aprendiz - baseado no modelo do aluno, *profile* cognitivo e desempenho do aprendiz (acertos/erros), atualiza crenças e modelo do aluno. Conteúdo da mensagem KQML: • usuário • matéria • código capítulo • operação (próximo/anterior/fim) • desempenho (habilidades/deficiências)	(Mensagem Tática Pedagógica) Envia uma tática a ser aplicada - conforme o modelo do aluno, e *profile* cognitivo, o agente diagnóstico envia uma tática a ser utilizada para a aprendizagem do aluno. Conteúdo da mensagem KQML enviada ao agente mediador: • usuário • matéria • tática (qual a tática a ser utilizada) • preferências usuário • nível do usuário (superior, inferior ou moderado)

A TABELA 5.1, assim como a TABELA 5.2 e TABELA 5.3, representam no lado esquerdo a *mensagem de entrada* e no lado direito a *mensagem de saída*, especificando os respectivos parâmetros do conteúdo das mensagens KQML trocadas entre os agentes.

TABELA 5.2 - Mensagens do Agente Mediador

Conteúdo de Entrada	Conteúdo de Saída
(Mensagem Ação Usuário) Recebendo Ação Usuário – conforme a ação do usuário, o agente mediador vai repassar uma mensagem Comportamento do Aluno para o agente diagnóstico, ou finalizar a sessão do usuário, gerando uma mensagem Requisição Conteúdo Pedagógico com tipo operação igual a *fim*. Possui o seguinte conteúdo: • usuário • operação (próximo/anterior/fim) • resposta exercício (pode ter resposta do exercício para que possa verificar se usuário acertou ou errou) (Mensagem Táticas Pedagógicas) Recebendo uma tática a ser aplicada vinda do agente diagnóstico. Possui o seguinte conteúdo de mensagem KQML: • usuário • matéria • tática (qual a tática a ser utilizada) • preferências usuário • nível do usuário (superior, inferior ou moderado) (Mensagem Conteúdo Pedagógico) Recebe o conteúdo da mensagem KQML em forma de HTML, gerado dinamicamente pelo agente semiótico. Possui o seguinte conteúdo: • matéria • código capítulo (é interessante repassar o código principal do conteúdo para que o agente mediador possa repassá-lo para o agente diagnóstico, o qual por sua vez, pode repassar para o agente social/colaborativo para discutir sobre determinado conteúdo) • conteúdo (propriamente dito, em forma de HTML)	(Mensagem Requisição Conteúdo Pedagógico) Conforme a tática solicitada pelo agente diagnóstico e a preferência do aprendiz, vai solicitar para o agente semiótico que construa o conteúdo a ser mostrado ao aprendiz. Possui o seguinte conteúdo de mensagem KQML: • usuário que solicitou • matéria • tática (apresentação de conteúdo, podendo ser um tipo específico como páginas HTML, citações, exemplos, *links*, palavras-chave ou exercícios) • operação (próximo/anterior/fim) • preferências do usuário • nível do usuário (superior, inferior ou moderado) • prioridade (Mensagem Login Usuário) Envia ao agente semiótico o *login* do usuário. Possui o seguinte conteúdo KQML: • usuário • senha • matéria (Mensagem Comportamento do Aluno) Enviando o comportamento do aprendiz em relação a uma ação feita pelo usuário. Possui o seguinte conteúdo KQML: • usuário • matéria • código capítulo • operação (próximo/anterior/fim) • desempenho (habilidades/deficiências) (Mensagem Ações do Agente Mediador) Envia mensagens para o usuário de forma a apresentar o conteúdo, incentivando o aluno, motivando e exibindo emoções.

TABELA 5.3 - Mensagens do Agente Semiótico

Conteúdo de Entrada	Conteúdo de Saída
(Mensagem Requisição Conteúdo Pedagógico) **Vinda do agente mediador** - Caso seja a operação *próximo*, busca na base de dados o próximo conteúdo pedagógico a ser mostrado, conforme a tática especificada. Se não existir próximo conteúdo, retorna uma mensagem com conteúdo do item vazio, para que o agente mediador (e conseqüentemente agente diagnóstico) decida qual o próximo passo (qual a próxima tática a ser utilizada). De forma similar funciona quando a operação for *anterior*. Caso seja operação *fim*, guarda no banco de dados a última ação feita pelo aprendiz. A mensagem KQML possui o seguinte conteúdo: • usuário que solicitou • matéria • tática (apresentação de conteúdo, podendo ser um tipo específico como páginas HTML, citações, exemplos, *links*, palavras-chave ou exercícios) • operação (próximo/anterior/fim) • preferências do usuário • nível do usuário (superior, inferior ou moderado) • prioridade (Mensagem Requisição Conteúdo Pedagógico) **Vinda do agente colaborativo** - verifica se determinado padrão reconhecido pelo agente colaborativo, durante assistência em ferramenta de colaboração, faz parte de determinado conteúdo. Esta mensagem KQML possui o seguinte conteúdo: • matéria • código capítulo • padrão reconhecido • prioridade (Mensagem Login Usuário) O agente semiótico verifica na base de dados se o usuário está cadastrado. Se estiver, mostra o último conteúdo pedagógico acessado pelo usuário. Se não estiver cadastrado, cadastra e mostra o 1º conteúdo pedagógico. Conteúdo da mensagem KQML: • usuário • senha • matéria	(Mensagem Conteúdo Pedagógico) **Destino agente mediador** - Envia o conteúdo pedagógico, para o agente mediador, gerado dinamicamente a partir da seleção dos signos ideais a serem apresentados em forma de página HTML. Caso não exista conteúdo/ exercício apropriado (anterior ou próximo), envia mensagem KQML com conteúdo vazio: • matéria • código capítulo (é interessante repassar o código principal do conteúdo para que o agente mediador possa repassá-lo para o agente diagnóstico, o qual por sua vez, pode repassar para o agente social/colaborativo para discutir sobre determinado conteúdo) • conteúdo (propriamente dito, em forma de HTML) (Mensagem Conteúdo Pedagógico) **Destino agente colaborativo** – Envia a resposta se determinado padrão reconhecido pelo agente colaborativo faz parte de determinado conteúdo (código capítulo). Conteúdo da mensagem KQML: • matéria • código capítulo (é interessante repassar o código principal do conteúdo para que o agente colaborativo possa referenciá-lo futuramente) • resposta reconhecimento padrão

5.2 Regras de comportamento

Todas as ações são executadas como resultado de estímulos gerados através de mensagens vindas dos agentes mediadores ou agentes colaborativos existentes na sociedade. Regras de comportamento determinam o curso da ação que um agente deve tomar em todo o ponto do início ao fim da execução do agente.

Regras de comportamento podem ser vistas como WHEN-IF-THEN, segundo Bica [BIC 99].

A parte WHEN da regra endereça um novo evento ocorrendo no ambiente do agente e inclui novas mensagens recebidas de outros agentes. O IF compara as condições do ambiente com as condições requeridas para que a regra seja aplicada. A parte THEN define as ações do agente incluindo ações de comunicação e ações pessoais.

Exemplo de formato de regras de comportamento, utilizadas na implementação, pode ser visualizada na FIGURA 5.1:

```
NAME nome regra
WHEN
        Condições das Mensagens
IF
        Condições do Ambiente
THEN
        Ações Pessoais
        Ações de Comunicação
```

FIGURA 5.1 – Formato de Regras de Comportamento

As regras de comportamento utilizadas na implementação do agente semiótico, funcionam da seguinte maneira:

- **Login_Usuário**: esta regra acontece quando o agente mediador envia uma mensagem ao agente semiótico, informando que um aprendiz se conectou ao sistema. O agente semiótico verifica na base de dados se o usuário está cadastrado. Se estiver, mostra o último conteúdo pedagógico acessado pelo usuário, desencadeando a regra Conteúdo_Pedagógico. Se não estiver cadastrado, cadastra o aprendiz e mostra o primeiro conteúdo pedagógico pela regra Conteúdo_Pedagógico;
- **Conteúdo_Pedagógico**: o agente semiótico envia uma mensagem para o agente mediador como resposta da regra Requisição_Conteúdo_Pedagógico ou Login_Usuário. O agente semiótico realiza uma consulta na base de dados para encontrar os signos necessários para montar o material instrucional a ser apresentado ao aluno. Para montar esta seqüência de signos, ele segue os exemplos apresentados na seção 4.2. Caso não encontre o signo adequado (dependendo do tipo operação *anterior*/*próximo*), retorna uma mensagem com conteúdo vazio, para que o agente mediador repasse-a ao agente diagnóstico, o qual vai decidir por uma nova tática a ser aplicada no ensino-aprendizagem do aluno. Caso seja operação *fim*, guarda no banco de dados a última ação feita pelo aprendiz. Se esta mensagem for enviada pelo agente colaborativo, o agente semiótico vai consultar a base de dados a fim de verificar se o padrão reconhecido pelo agente colaborativo faz parte do conteúdo/signo que está sendo tratado no momento (pela ferramenta de colaboração utilizada pelo

agente colaborativo), retornando para o agente colaborativo o resultado desta verificação;

- **Requisição_Conteúdo_Pedagógico**: esta regra é disparada pelo agente mediador para o agente semiótico ou do agente colaborativo ao agente semiótico. No primeiro caso, conforme a tática definida pelo agente diagnóstico (regra Táticas_Pedagógicas) e a preferência do aprendiz (regra Ação_Usuário), o agente mediador vai enviar uma mensagem ao agente semiótico solicitando que o mesmo gere um conteúdo pedagógico a ser apresentado ao aprendiz (regra Conteúdo_Pedagógico). No segundo caso, o agente colaborativo vai solicitar ao agente semiótico que verifique se determinado padrão, encontrado durante interação na ferramenta de colaboração, faz parte de determinado conteúdo pedagógico de que se trata no momento.

Exemplos de regras a serem utilizadas pelos outros agentes são:

- **Comportamento_do_Aluno**: o agente mediador informa ao agente diagnóstico o comportamento do aprendiz como resultado da interação no sistema, incluindo o desempenho (habilidades/deficiências e acertos/erros) do aprendiz, e qual ação (anterior/próximo/fim) executada pelo aluno;
- **Táticas_Pedagógicas**: o agente diagnóstico baseado no modelo do aluno (*profile* cognitivo), desempenho do aprendiz (acertos/erros), e ação do aluno (anterior/próximo/fim), atualiza crenças e modelo do aluno, enviando uma nova tática de ensino-aprendizagem do aluno para ser utilizada pelo agente mediador;
- **Ação_Usuário**: conforme a ação do usuário, o agente mediador vai decidir se deve mostrar um conteúdo (próximo/anterior), ou finalizar a sessão do usuário (fim) através da regra Requisição_Conteúdo_Pedagógico;
- **Ações_do_Agente_Mediador**: Envia mensagens para o usuário de forma a apresentar o conteúdo, incentivando o aluno, motivando-o e exibindo emoções através de um agente animado.

5.3 Base de Dados

O armazenamento dos signos é feito em um banco de dados, por exemplo Oracle, onde a definição das tabelas é mostrado a seguir, possuindo as seguintes idéias:

- uma matéria pode ter um ou mais itens, onde estes itens podem ser do tipo: capítulo ou seção/subseção (isto é identificado pelo campo codigo_pai na tabela item), parágrafo, arquivo HTML, figura, tabela, lista, exemplo, citação, *link* para assuntos referentes ao conteúdo em questão, palavras-chave (que serão utilizadas pelo agente colaborativo no reconhecimento de padrões) e/ou exercícios;
- os itens do tipo capítulo e seção possuem grau de dificuldade (fácil, médio ou difícil);
- um item do tipo exercício é composto por enunciado, resposta e possivelmente itens deste exercício;
- uma matéria pode possuir, ainda, referências bibliográficas.

A FIGURA 5.2 a seguir mostra como está a estrutura das tabelas na base de dados.

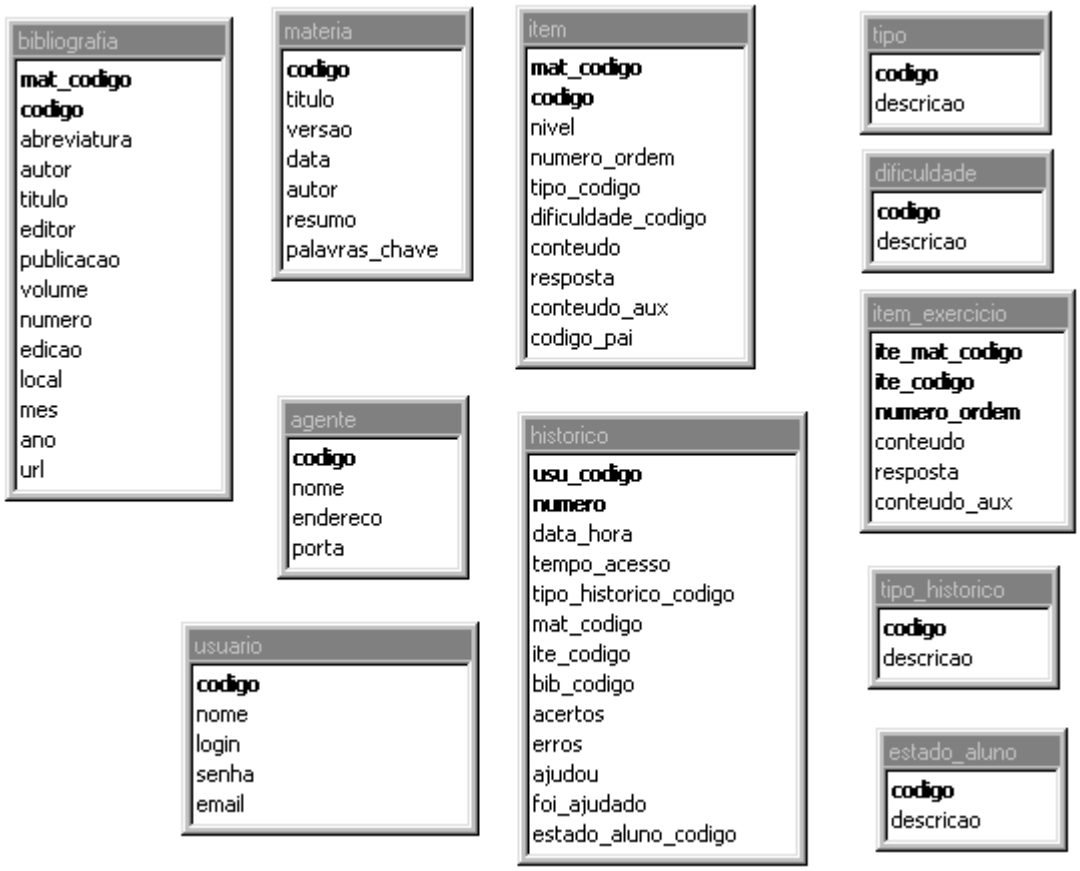

FIGURA 5.2 – Estrutura da Base de Dados

A seguir vamos especificar as tabelas e seus respectivos campos, onde PK (*primary key*) significa que o campo faz parte da chave primária da tabela, representado pelo símbolo (#); e NN (*not null*) significa que o campo é de preenchimento obrigatório, representado pelo símbolo (*).

A tabela Materia especificada na TABELA 5.4 guarda as informações gerais sobre a matéria/conteúdo a ser apresentado para o aluno.

TABELA 5.4 - MATERIA

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	codigo	number		código identificador da matéria
	*	titulo	varchar2	100	título da matéria
	*	versao	varchar2	50	guarda a versão da matéria
	*	data	date		data criação
	*	autor	varchar2	100	informação do autor da matéria
	*	resumo	varchar2	255	guarda um resumo da matéria
	*	palavras_chave	varchar2	255	guarda palavras-chave da matéria

A tabela Item especificada na TABELA 5.5 guarda as informações sobre os itens da matéria, cujo item pode ser: capítulo/seção, parágrafo, arquivo HTML, figura, tabela, lista, exemplo, citação, *link*, palavras-chave, exercícios (múltipla-escolha, dissertativa, verdadeiro ou falso, relacionar colunas, completar lacunas e ordenação).

TABELA 5.5 - ITEM

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	mat_codigo	number		código identificador da matéria
#	*	codigo	number		código identificador do item
	*	nivel	number		identifica o nível (1,2,3) do capítulo, podendo ser (capítulo, seção, subseção) e assim, sucessivamente, quantos níveis se desejar. Obs.: não precisa ser chave, uma vez que o código do item/capítulo para aquela matéria será único
	*	numero_ordem	number		identifica a ordem em que vão aparecer os itens num mesmo nível (em que ordem vão aparecer os capítulos/seções, parágrafos, *links*, etc.
	*	tipo_codigo	number		chave-estrangeira da tabela tipo representando os tipos possíveis de item: capítulo/seção, parágrafo, arquivo HTML, figura, tabela, lista, exemplo, citação, *link*, palavras-chave, exercícios (múltipla-escolha, dissertativa, verdadeiro ou falso, relacionar colunas, completar lacunas e ordenação)
		dificuldade_codigo	number		chave-estrangeira da tabela dificuldade, representando a dificuldade (fácil, médio, difícil) do capítulo/seção/subseção
	*	conteudo	varchar2	65535	o conteúdo do item, podendo ser o conteúdo propriamente dito do parágrafo, HTML, *link*, etc., ou guarda a informação do título do capítulo/seção
		codigo_pai	number		identifica o código do item pai, por exemplo o código do capítulo para uma seção, ou o código da seção para uma subseção, ou ainda a qual capítulo faz parte tais parágrafos, citações, etc.

A tabela Tipo especificada na TABELA 5.6 guarda as informações sobre os tipos de itens que pode conter a tabela Item.

TABELA 5.6 - TIPO

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	codigo	number		código identificador do tipo
	*	descricao	varchar2	100	descrição representando os tipos possíveis de item: capítulo/seção, parágrafo, arquivo HTML, figura, tabela, lista, exemplo, citação, *link*, palavras-chave, exercícios (múltipla-escolha, dissertativa, verdadeiro ou falso, relacionar colunas, completar lacunas e ordenação)

A tabela Dificuldade especificada na TABELA 5.7 guarda as informações sobre os tipos de dificuldade usadas na tabela Item.

TABELA 5.7 - DIFICULDADE

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	codigo	number		código identificador da dificuldade
	*	descricao	varchar2	50	descrição das dificuldades (por exemplo: fácil, médio e difícil)

A tabela Item_exercicio especificada na TABELA 5.8 guarda as informações sobre os itens do exercício.

TABELA 5.8 - ITEM_EXERCICIO

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	ite_mat_codigo	number		código identificador da matéria
#	*	ite_codigo	number		código identificador do item
#	*	numero_ordem	number		ordem dos itens do exercício
	*	conteudo	varchar2	200	conteúdo do item do exercício
	*	resposta	varchar2	200	dependendo do tipo de exercício, podem ter respostas individuais para cada item do exercício
		conteudo_aux	varchar2	200	utilizado, por exemplo, quando o item requer dois conteúdos (por exemplo, relacionar colunas, precisa uma coluna sendo a original e a outra a ser relacionada)

A tabela Historico especificada na TABELA 5.9 armazena todos os acessos dos alunos no sistema, seja entrada, saída ou conteúdo que tenha acessado.

TABELA 5.9 - HISTORICO

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	usu_codigo	number		código identificador do usuário
#	*	numero	number		número de controle incremental para identificar um histórico
	*	data_hora	date		data em que ocorreu alguma ação
		tempo_acesso	varchar2	50	tempo de acesso
	*	tipo_historico_codigo	number		chave-estrangeira da tabela tipo_historico, podendo representar entrada, saída ou acesso a um conteúdo
		mat_codigo	number		código da matéria acessada (se for o caso)
		ite_codigo	number		código do item acessado (se for o caso)
		bib_codigo	number		código da bibliografia (caso acessado)
		acertos	number		número de acertos (no caso de exercícios)
		erros	number		número de erros (no caso de exercícios)
		ajudou	boolean		informa se ajudou alguém
		foi_ajudado	boolean		informa se foi ajudado por alguém
		estado_aluno_codigo	number		chave-estrangeira da tabela estado_aluno representando o estado do aluno no momento

A tabela Tipo_historico especificada na TABELA 5.10 guarda as informações sobre os tipos de histórico que podem ser usados na tabela Historico.

TABELA 5.10 - TIPO_HISTORICO

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	codigo	number		código identificador do tipo
	*	descricao	varchar2	100	os tipos de histórico possíveis representam as seguintes ações do usuário: entrada, saída, acesso a item de conteúdo, ou acesso a bibliografia

A tabela Bibliografia especificada na TABELA 5.11 guarda as informações da bibliografia da matéria.

TABELA 5.11 - BIBLIOGRAFIA

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	mat_codigo	number		código identificador da matéria
#	*	codigo	number		código identificador da bibliografia
	*	abreviatura	varchar2	50	abreviatura da bibliografia, podendo ser por exemplo: JUN01 ou Jung, 2001
	*	autor	varchar2	255	autor da bibliografia
	*	titulo	varchar2	255	título da bibliografia
		editor	varchar2	100	editor, por exemplo, dos anais
		publicacao	varchar2	100	publicação/evento
		volume	varchar2	10	volume
		numero	varchar2	10	numero
		edicao	varchar2	50	edição
		local	varchar2	100	local do evento
		mes	varchar2	20	mês
	*	ano	number	4	ano
		url	varchar2	255	*url*

A tabela Usuario especificada na TABELA 5.12 guarda as informações do usuário.

TABELA 5.12 - USUARIO

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	codigo	number		código identificador do usuário
	*	nome	varchar2	100	nome do usuário
	*	login	varchar2	50	nome usado no *login*
	*	senha	varchar2	50	senha do usuário
		email	varchar2	255	email do usuário
		estado_aluno_codigo	number		chave-estrangeira da tabela estado_aluno (indicando o estado em que se encontra o aluno: estado inferior, estado moderado, estado superior)

A tabela Estado_aluno especificada na TABELA 5.13 guarda os tipos possíveis de estados dos alunos.

TABELA 5.13 - ESTADO_ALUNO

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	codigo	number		código identificador do estado do aluno
	*	descricao	varchar2	50	nome dos possíveis estados dos alunos (indicando o estado em que se encontra o aluno: inferior, moderado, superior)

A tabela Agente especificada na TABELA 5.14 guarda as informações sobre os agentes.

TABELA 5.14 - AGENTE

PK	NN	Nome	Tipo	Tamanho	Descrição
#	*	codigo	number		código identificador do agente
	*	nome	varchar2	50	nome do agente
	*	endereco	varchar2	50	endereço do agente
	*	porta	number		número da porta para acessar o agente

Este modelo comporta várias matérias/conteúdos, portanto, poderá ser utilizado para o desenvolvimento de aplicações de EAD em diversos domínios do conhecimento. Pode ser utilizado, também, para outras propostas/implementações de ambientes de ensino-aprendizagem que comportem esta modelagem.

O modelo descrito é simples mas suficiente para as necessidades atuais do sistema proposto pelo grupo.

5.4 Ambiente de Gerência do Material Instrucional

Para que os signos possam ser inseridos na base de conhecimento, foi construída uma interface feita em Java, onde levou-se em conta alguns aspectos sugeridos por Leite [LEI 2001]:

- utilização de ícones e mensagens sensíveis ao contexto (por exemplo, mensagens quando o mouse passa sobre um botão ou uma caixa de texto);
- sempre que o aluno faz uma ação, ter uma resposta para a ação feita, seja visual (por exemplo, uma nova tela aparecendo), ou através de mensagem (por exemplo, mensagem de que foi salvo com sucesso);
- utilização de signos, ícones e símbolos de fácil utilização (onde símbolos são signos já conhecidos e padronizados, por exemplo, o ícone de abrir ou salvar); e de maneira iterativa e evolutiva (conforme vai interagindo com o sistema, as telas e ícones vão sendo apresentados coerentemente);
- respeitar a idéia da ação a ser feita na tela de forma similar, por exemplo, como escrevemos habitualmente em um papel (começando de cima para baixo, e da esquerda para a direita);
- respeitar símbolos já padronizados como os três pontinhos após uma palavra, como por exemplo “abrir...”, representa que irá aparecer outra tela de configuração do arquivo a ser aberto;
- não deixar botões (por exemplo do tipo gravar/salvar) muito longe da ação que o usuário está fazendo.

Dentre as qualidades de um sistema web proposto por Leite [LEI 2001], cita-se as que este sistema procurou apresentar:

- funcional
- eficiente
- robusto e confiável
- bem documentado
- manutenível
- testável
- portável
- reutilizável
- interoperável
- fácil de usar
- legível
- atrativo
- organizado
- correto

- atualizado
- adequado aos usuários
- adequado à tecnologia
- adequado ao propósito.

Além disso, o sistema teve em mente o modelo de Norman [NOR 86, apud in LEI 2001] mostrado na FIGURA 5.3, onde tem-se o golfo da execução (o usuário tem a intenção de fazer algo, verifica quais as ações ou seqüência de ações ele deve executar, e executa) e o golfo da avaliação (o usuário percebe o resultado, interpreta o resultado e avalia se realmente era o esperado/desejado). Dessa forma, para toda a ação no sistema (Golfo de Execução), existe um resultado (Golfo da Avaliação).

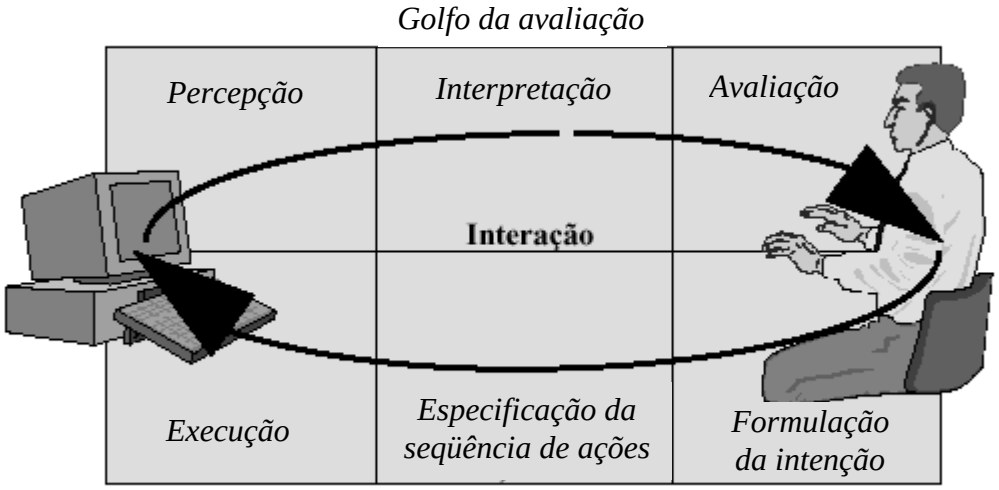

FIGURA 5.3 – Modelo de Norman [LEI 2001]

Assim, temos como resultado, por exemplo, a tela do sistema mostrado a seguir:

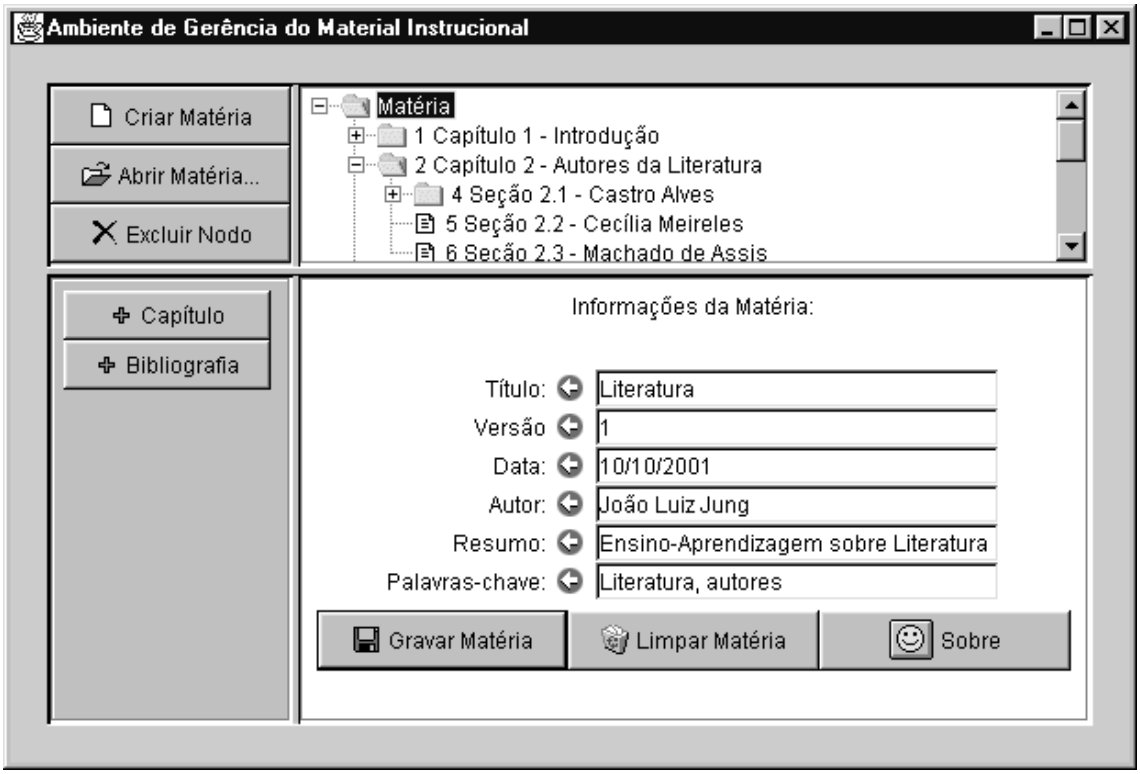

FIGURA 5.4 – Tela do Ambiente de Gerência do Material Instrucional

Pode-se observar, no *layout* da tela, heurísticas em termos do posicionamento das ações a serem feitas: de cima para baixo, da esquerda para a direita (primeiro criar matéria, depois abrir matéria já existente, e depois deletar um nodo da matéria). De forma similar, na parte de baixo à esquerda, pode-se, uma vez existindo a matéria, inserir novos capítulos ou bibliografias relacionadas à matéria.

Observa-se também a utilização de signos, ícones e símbolos já padronizados e de fácil conhecimento e utilização para o usuário.

Além disso, o ambiente procurou seguir as heurísticas sugeridas por Gomes e Vicari [GOM 99] conforme é mostrado abaixo, onde o sistema deve ser:

- eficiente – minimizando o esforço gasto para executar uma tarefa;
- conveniente – as operações podem ser executadas de muitas maneiras, por exemplo, o usuário pode navegar pela árvore de conteúdo e atualizar ou adicionar novos conteúdos, ou pode também navegar pelos botões disponíveis no lado esquerdo parte inferior para adicionar novos conteúdos;
- auto-descritivo – o ícones utilizados facilitam a utilização do sistema;
- prestativo – ao passar com o mouse sobre um objeto na tela, automaticamente aparece um rótulo explicativo;
- confortável – por exemplo, permite o usuário dimensionar a largura e altura dos *frames* que dividem a tela em quatro partes, conforme o seu gosto;
- consistente – as ações para adicionar e navegar pelos conteúdos do material instrucional são semelhantes em qualquer nível;
- obediente – permite ao usuário acessar qualquer nível e tipo de conteúdo a qualquer momento, processando assim as vontades do usuário;
- códigos e denominações significativos – os símbolos e ícones utilizados são de significado coerente com as ações;
- complacente – existe o botão voltar para retornar ao ponto anterior, caso seja interesse do usuário, ou ainda a opção de excluir nodo caso não se queira mais um determinado item;
- tolerante a falhas – existem mecanismos para evitar e prevenir erros, por exemplo, mensagens perguntando se realmente deseja deletar nodos da árvore, ou ícones representando a obrigatoriedade do preenchimento do campo;
- mensagens de erro – explica o que aconteceu de errado, por exemplo, quando deixou de preencher um campo obrigatório ao salvar um determinado conteúdo;
- passivo – em termos de que o usuário pode interromper ou mudar de operação a qualquer momento;
- fornecer *feedback* – sempre que o usuário executa uma operação, existe um retorno, seja visual ou em forma de mensagem sobre o que foi executado.

Nas próximas figuras, procurou-se demonstrar algumas heurísticas apresentadas pelo sistema.

Por exemplo, na figura a seguir, posicionando o mouse sobre o botão “Capítulo”, aparece um rótulo explicativo sobre a função do botão, demonstrando a característica do sistema ser prestativo. Pode-se visualizar, também, que os ícones utilizados são auto-descritivos com códigos e denominações significativos.

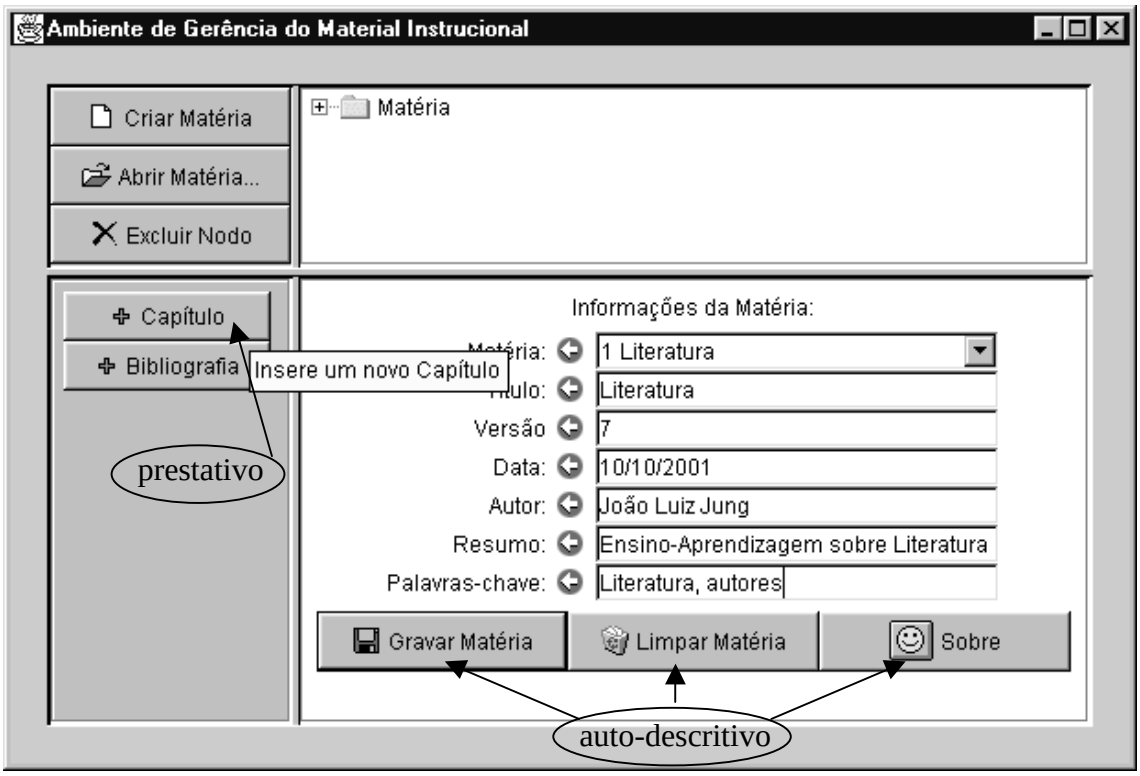

FIGURA 5.5 – Sistema prestativo e auto-descritivo

Ao pressionar o botão que insere um novo Capítulo, tem-se a tela representada pela figura a seguir, onde é possível visualizar que o sistema é complacente (existe um botão “Voltar” que permite desfazer a ação feita).

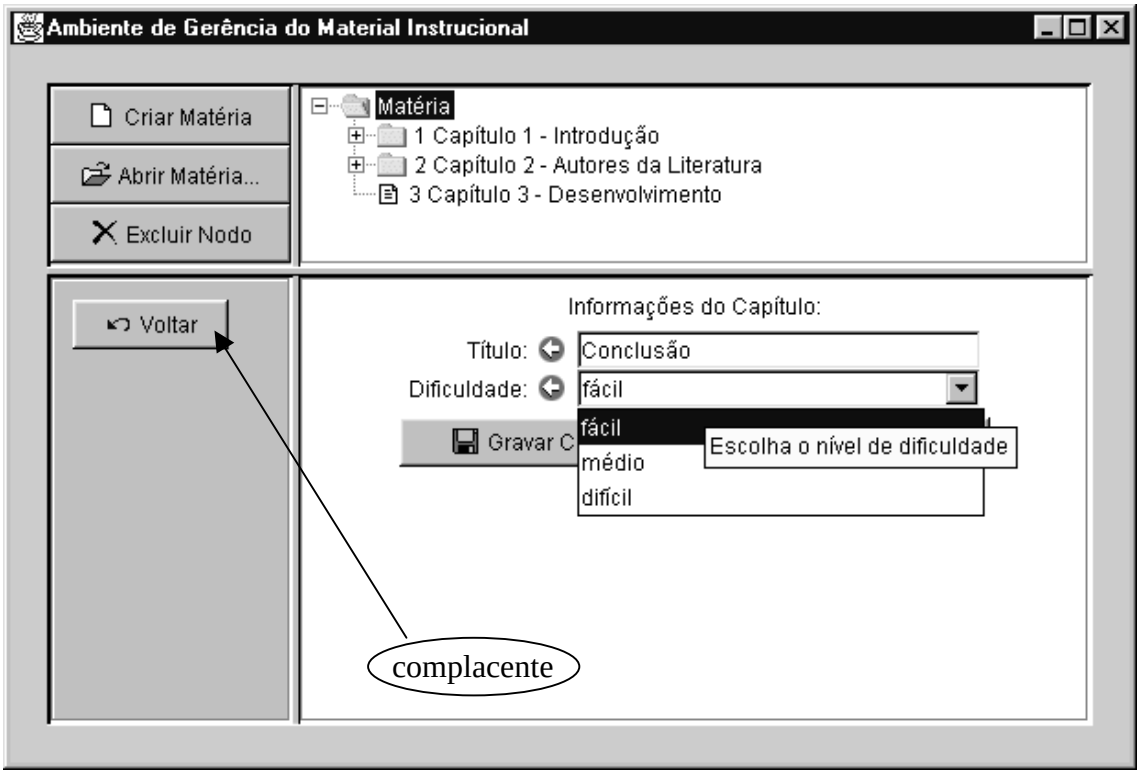

FIGURA 5.6 – Sistema complacente

Ao pressionar o botão “Gravar Capítulo”, o sistema fornece um *feedback* sobre o que foi executado, representado, na figura abaixo, através de um retorno visual em forma de mensagem.

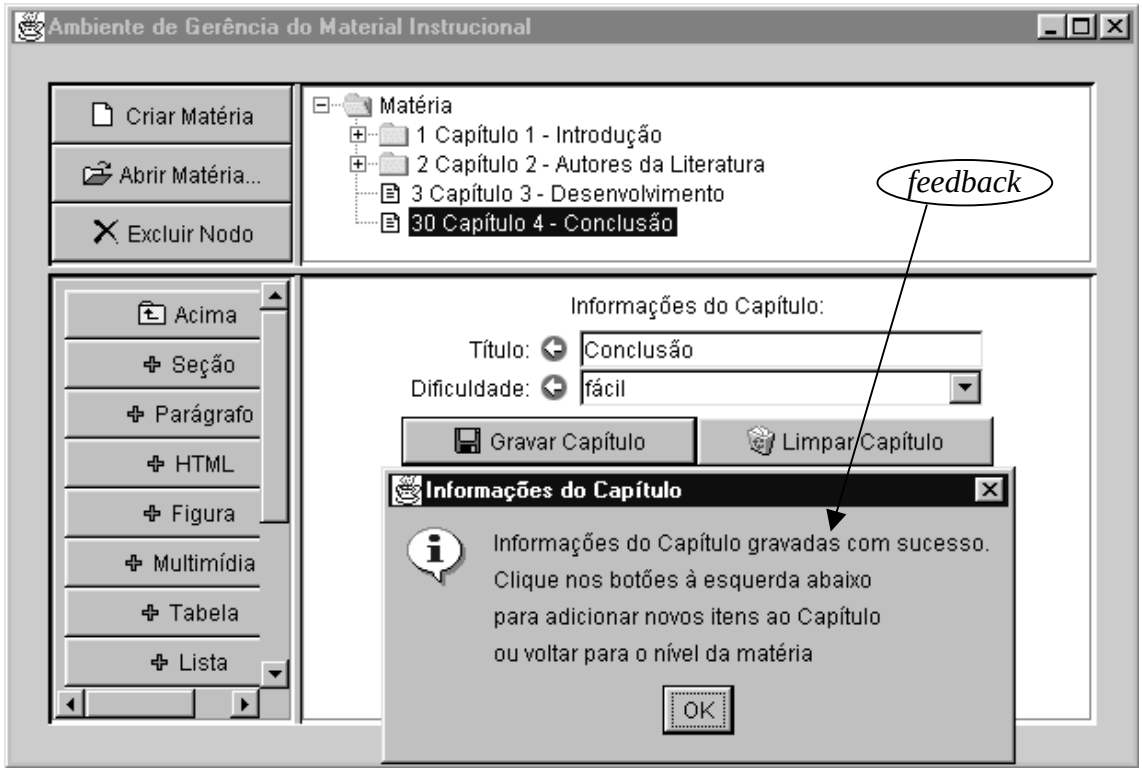

FIGURA 5.7 – *Feedback* do Sistema

Na figura abaixo, ao pressionar o botão “Excluir Nodo”, o sistema apresenta a característica de ser tolerante a falhas, prevenindo e evitando erros.

FIGURA 5.8 – Sistema tolerante a falhas

Na figura a seguir, pode-se visualizar que o sistema é confortável, permitindo ao usuário dimensionar a largura e altura dos frames.

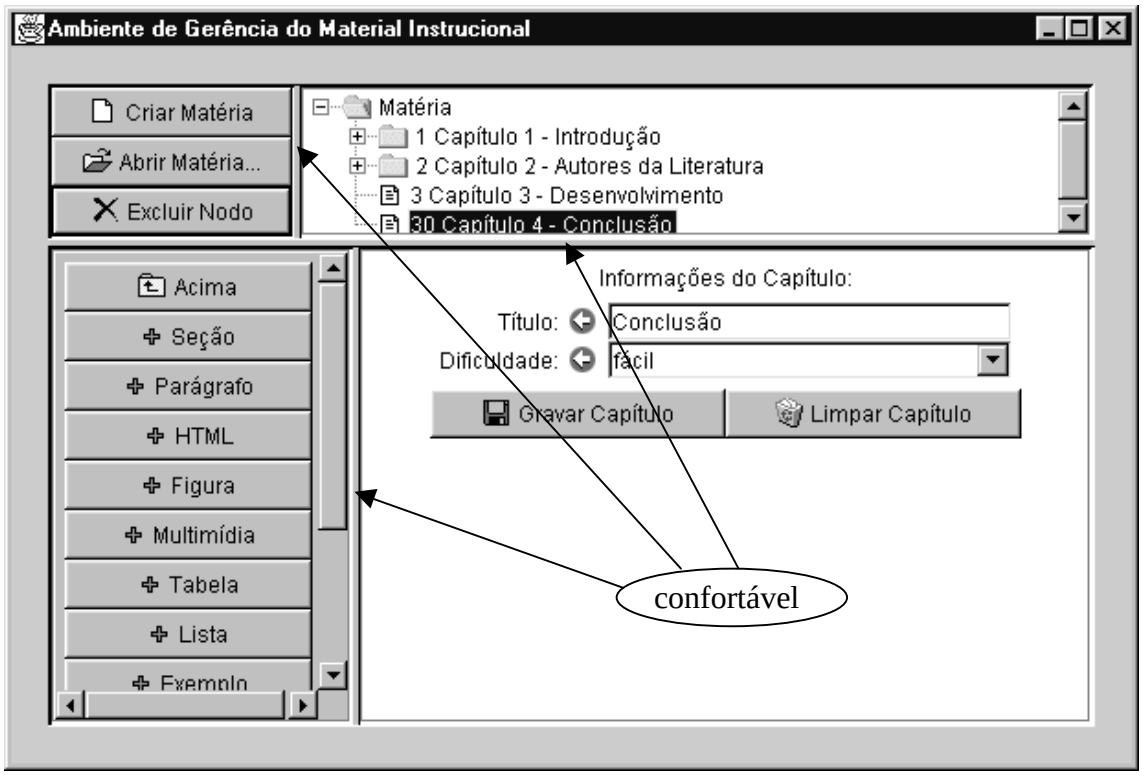

FIGURA 5.9 – Sistema confortável

5.5 Ambiente de Educação a Distância

Nas figuras a seguir, apresenta-se o ambiente de educação a distância, simulando algumas trocas de mensagens KQML entre os agentes no sistema, conforme apresentado na seção 5.1.

Por exemplo, o agente mediador envia uma mensagem KQML Login Usuário para o agente semiótico como mostra a tabela abaixo, desencadeada pela ação do usuário na FIGURA 5.10.

TABELA 5.15 - Mensagem KQML Login Usuário

Parâmetro	Valor
:performative	Tell
:sender	agente mediador
:receiver	agente semiótico
:ontology	login usuário
:in-reply-to	agente mediador
:reply-with	conteúdo pedagógico
:content	• usuário • senha • matéria

FIGURA 5.10 – Login Usuário

O agente semiótico ao receber a mensagem da TABELA 5.15, desencadeia as regras de comportamento Login_Usuário e Conteúdo_Pedagógico, conforme visto na seção 5.2, retornando ao agente mediador a mensagem KQML Conteúdo Pedagógico mostrada na TABELA 5.16 abaixo.

TABELA 5.16 - Mensagem KQML Conteúdo Pedagógico

Parâmetro	Valor
:performative	Tell
:sender	agente semiótico
:receiver	agente mediador
:ontology	conteúdo pedagógico
:content	• matéria • código capítulo • conteúdo (material instrucional em forma de HTML)

O resultado desta mensagem é representado pela FIGURA 5.11, onde tem-se a construção dinâmica do material instrucional pelo agente semiótico, baseando-se no exemplo de utilização da LEMD, conforme visto na seção 4.2. Este resultado é enviado para o agente mediador e, conseqüentemente, para o usuário.

Pode-se observar, na figura, a padronização dos signos (por exemplo: capítulo, seção, parágrafo, exemplo, citação, lista), gerando assim um padrão cognitivo com o objetivo de facilitar a usabilidade do sistema e auxiliar o processo mnemônico de aprendizado do aluno.

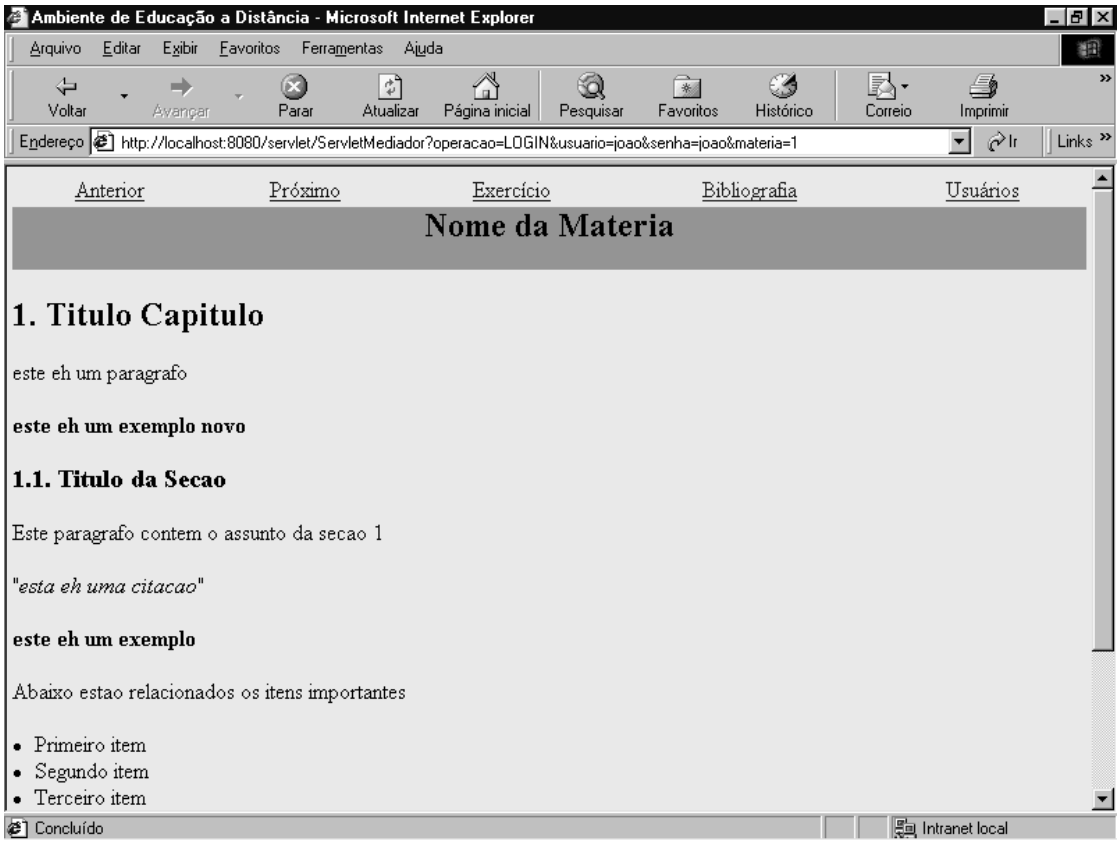

FIGURA 5.11 – Conteúdo Pedagógico

6 Conclusões e Trabalhos Futuros

O uso de SMA em STI permite uma melhor representação do domínio, com uma maior possibilidade de táticas pedagógicas, uma melhor representação do aluno espelhado no modelo do aluno construído no sistema, além da construção de tutores mais robustos e eficientes que possam auxiliar na aprendizagem.

O campo de agentes autônomos e sistemas multiagentes é uma área instigante em rápida expansão de pesquisa e desenvolvimento. Em seu núcleo, está o conceito de agentes autônomos interagindo com os outros para seu benefício individual e/ou coletivo. Esta estrutura conceitual básica tem-se tornado de uso geral em uma variedade de disciplinas (computação distribuída, sistemas orientados a objeto, engenharia de software, inteligência artificial, economia, sociologia, e ciência organizacional), e oferece um poderoso *framework* de análise, projeto, e implementação em uma diversificada gama de soluções de software.

Procurou-se aqui utilizar a Engenharia Semiótica através do formalismo da LEMD para geração de signos, ícones e símbolos, representando o material instrucional a ser apresentado ao aprendiz. A função de geração dos signos apropriados é responsabilidade do agente semiótico, respeitando assim, o importante papel que representa o processo mnemônico de aprendizado do aluno através dos signos, inspirado em Vygotsky.

Além disso, procurou-se iniciar a construção de um ambiente de STI, implementado como um ambiente de sistema multiagente. Esta sociedade de agentes proporciona um ambiente que possibilita, através da interação-social dos agentes artificiais e humanos (tutores e aprendizes), um processo de ensino-aprendizado inspirado nas idéias defendidas por Vygotsky, a chamada Teoria Sócio-Interacionista.

O modelo da base de dados desenvolvido comporta várias matérias/conteúdos, podendo ser utilizado para o desenvolvimento de aplicações de EAD em diversos domínios do conhecimento, além de poder ser utilizado, também, para outras propostas ou implementações de ambientes de ensino-aprendizagem que comportem esta modelagem.

Embora os demais agentes ainda não estejam implementados, este agente foi testado, através de simulações das trocas de mensagens com os agentes colaborativo e mediador.

Permitir a definição da quantidade de ajuda que um determinado aluno gostaria de receber, seja de agentes artificiais ou humanos (tutores e aprendizes), ou quanto um aluno se dispõe para ajudar, de forma que o próprio aluno possa ser uma espécie de tutor ou mentor, como sugere Bull [BUL 2000], são trabalhos futuros que podem ser efetivados levando-se em conta a existência da interação social no ambiente. Estas interações sociais entre o aluno e agentes artificiais ou humanos (tutores e aprendizes) em forma de ajuda trocada entre eles, por exemplo, aluno ajudando ou recebendo ajuda (pergunta/resposta), pode ser transformada e armazenada na base de dados no formato de perguntas mais freqüentes (FAQ), a serem utilizadas como material instrucional no processo de ensino-aprendizado.

Uma ferramenta de configuração dos diferentes estilos a serem utilizados na apresentação do material instrucional, permitindo a alteração dos arquivos XSL, é outro trabalho futuro que pode ser implementado.

Além disso, pode-se criar um ambiente de análise do histórico das ações realizadas pelos alunos. Estas ações podem ser individuais, mediadas pelo agente mediador, e neste caso, poderia-se apresentar gráficos representando o tempo de acesso/permanência nas páginas, número de acertos/erros, e quantas vezes um aluno ajudou ou foi ajudado. Outra forma de ações está relacionada às trocas de mensagens entre os alunos na ferramenta de colaboração mediadas pelo agente colaborativo. Neste caso, o professor poderia analisar um log desta troca de mensagens.

Os exercícios existentes no Ambiente de Gerência do Material Instrucional, poderiam também ser utilizados como questões de provas a serem aplicadas pelo sistema no momento de avaliação do aluno.

Com o avanço da implementação dos demais agentes, poderemos verificar e analisar a usabilidade do agente semiótico, bem como avaliar os resultados obtidos com a utilização do sistema como um todo.

Anexo 1 Exemplo de Arquivo XML e XSL

Exemplo do arquivo **materia1.xml**:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<!DOCTYPE materia [
<!-- comentario * zero ou mais -->
<!-- comentario ? zero ou um -->

<!ELEMENT materia (titulo, capitulo*, bibliografia*)>
<!ATTLIST materia versao CDATA #REQUIRED
                  data   CDATA #REQUIRED
                  autor  CDATA #IMPLIED >
<!-- atributos devem ser usados quando info nao precisa ser mostrada-->

<!ELEMENT titulo (#PCDATA)>
<!ELEMENT codigo (#PCDATA)>

<!ELEMENT capitulo (codigo, titulo,
(secao|paragrafo|figura|tabela|exemplo|lista|citacao|link|hipertexto)*,
 exercicio*, bibliografia*)>
<!ATTLIST capitulo id CDATA #REQUIRED
                   nivel (1|2|3) "1" >

<!ELEMENT bibliografia (referencia*)>


<!ELEMENT referencia (autor, titulo, editor?, publicacao, volume?, numero?,
 edicao?, local?, mes?, ano, url?)>
<!ATTLIST referencia id CDATA #REQUIRED>
<!ELEMENT autor (#PCDATA)>
<!ELEMENT editor (#PCDATA)>
<!ELEMENT publicacao (#PCDATA)>
<!ELEMENT volume (#PCDATA)>
<!ELEMENT numero (#PCDATA)>
<!ELEMENT edicao (#PCDATA)>
<!ELEMENT local (#PCDATA)>
<!ELEMENT mes (#PCDATA)>
<!ELEMENT ano (#PCDATA)>
<!ELEMENT url (#PCDATA)>

<!ELEMENT secao (codigo, titulo,
(secao|paragrafo|figura|tabela|exemplo|lista|citacao|link|hipertexto)*)>
<!ATTLIST secao id CDATA #REQUIRED
          nivel  (1|2|3) "1" >

<!-- niveis 1, 2 e 3 de dificuldade -->

<!ELEMENT paragrafo (#PCDATA)>
<!ATTLIST paragrafo citacao CDATA #IMPLIED>
<!--citacao eh a abreviatura na referencia da bibliografia -->

<!ELEMENT figura (fig, codigo, nome)>
<!ATTLIST figura tipo (WMF|GIF|BMP|JPEG|TIFF) #REQUIRED
                 nome CDATA #REQUIRED>
<!ELEMENT fig EMPTY>
<!ELEMENT nome (#PCDATA)>

<!ELEMENT tabela (tab, codigo, nome)>
<!ELEMENT tab (#PCDATA)>

<!ELEMENT exemplo (#PCDATA)>
<!ELEMENT exemplo (ex, codigo, nome)>
<!ELEMENT ex (#PCDATA)>

<!ELEMENT lista (#PCDATA)>
<!ATTLIST lista tipo (C|T|N) "C" >
<!-- comentario circulo|traco|numero -->

<!ELEMENT citacao (#PCDATA)>
```

```
<!ELEMENT link (#PCDATA)>

<!ELEMENT hipertexto (#PCDATA)>

<!ELEMENT exercicio (tipo, codigo, enunciado, item*, resposta*)>
<!ATTLIST exercicio id CDATA #REQUIRED>
<!ELEMENT tipo (#PCDATA)>
<!ELEMENT enunciado (#PCDATA)>
<!ELEMENT item (#PCDATA)>
<!ELEMENT resposta (#PCDATA)>

]>
<materia versao="1.0" data="01/01/2001">
  <titulo>Nome da Materia</titulo>
  <capitulo id="1" nivel="1">
    <codigo>1</codigo>
    <titulo>Titulo Capitulo</titulo>
    <paragrafo>este eh um paragrafo</paragrafo>
    <exemplo>este eh um exemplo novo</exemplo>
    <secao id="1.1">
      <codigo>1.1</codigo>
      <titulo>Titulo da Secao</titulo>
      <paragrafo>Este paragrafo contem o assunto da secao 1</paragrafo>
      <citacao>esta eh uma citacao</citacao>
      <exemplo>este eh um exemplo</exemplo>
      <link>login.html</link>
      <paragrafo>Abaixo estao relacionados os itens importantes</paragrafo>
      <lista tipo="T">Primeiro item</lista>
      <lista tipo="T">Segundo item</lista>
      <lista tipo="T">Terceiro item</lista>
      <secao id="1.1.1" nivel="2">
        <codigo>1.1.1</codigo>
        <titulo>NIVEL=2Titulo da subsecao</titulo>
        <paragrafo>Texto da subsecao com nivel 2</paragrafo>
      </secao>
    </secao>
    <secao id="1.2" nivel="1">
      <codigo>1.2</codigo>
      <titulo>Titulo da secao seguinte</titulo>
      <paragrafo>Texto desta secao</paragrafo>
    </secao>
    <exercicio id="1">
      <tipo>Multipla Escolha</tipo>
      <codigo>1</codigo>
      <enunciado>Quais das opcoes abaixo esta correta</enunciado>
      <item>Opcao 1</item>
      <item>Opcao 2</item>
      <item>Opcao 3</item>
      <item>Opcao 4</item>
      <item>Opcao 5</item>
      <resposta>a</resposta>
    </exercicio>
    <exercicio id="2">
      <tipo>Multipla Escolha</tipo>
      <codigo>2</codigo>
      <enunciado>Escolha a alternativa correta</enunciado>
      <item>Item 1</item>
      <item>Item 2</item>
      <item>Item 3</item>
      <item>Item 4</item>
      <item>Item 5</item>
      <resposta>b</resposta>
    </exercicio>
  </capitulo>
  <capitulo id="2" nivel="1">
    <codigo>2</codigo>
    <titulo>Titulo Capitulo nivel 1</titulo>
    <secao id="2.1">
      <codigo>2.1</codigo>
      <titulo>Titulo da Secao</titulo>
      <paragrafo>Este paragrafo contem o assunto da secao 1 do capitulo 2
```

```
 nivel 1</paragrafo>
      <exemplo>este eh um exemplo</exemplo>
      <link>www.inf.ufrgs.br</link>
      <secao id="2.1.1" nivel="2">
        <codigo>2.1.1</codigo>
        <titulo>Titulo da subsecao nivel 2</titulo>
        <paragrafo>Texto da subsecao nivel 2</paragrafo>
      </secao>
    </secao>
  </capitulo>
  <bibliografia>
    <referencia id="MAR99">
      <autor>Maruyama, H. and Tamura, K. and Uramoto, N.</autor>
      <titulo>XML and Java: Developing of Web Applications</titulo>
      <publicacao>Addison-Wesley</publicacao>
      <local>MA</local>
      <mes>August</mes>
      <ano>1999</ano>
    </referencia>
    <referencia id="BRA00">
      <autor>Bradley, N.</autor>
      <titulo>The XML Companion</titulo>
      <publicacao>Addison-Wesley</publicacao>
      <edicao>2</edicao>
      <local>Great Britain</local>
      <mes>August</mes>
      <ano>2000</ano>
    </referencia>
    <referencia id="BRA01">
      <autor>Bradley, N.</autor>
      <titulo>The XML Companion</titulo>
      <publicacao>Addison-Wesley</publicacao>
      <edicao>3</edicao>
      <local>Great Britain</local>
      <mes>August</mes>
      <ano>2001</ano>
    </referencia>
    <referencia id="BOX00">
      <autor>Box, D. and Skonnard, A. and Lam, J.</autor>
      <titulo>Essential XML - Beyond Markup</titulo>
      <editor>Series</editor>
      <publicacao>Addison-Wesley</publicacao>
      <volume></volume>
      <numero></numero>
      <edicao></edicao>
      <local></local>
      <mes>July</mes>
      <ano>2000</ano>
      <url>http://www.develop.com/books/essentialxml</url>
    </referencia>
  </bibliografia>
</materia>
```

A idéia geral é que temos **matéria** e dentro de matéria podemos ter um **título**, zero ou mais **capítulos** e zero ou mais **bibliografias**, sendo que a matéria pode ser identificada por **versão**, **data** e **autor**. Um **capítulo** é constituído por um código, um título, zero ou mais seções, zero ou mais exercícios e zero ou mais bibliografias. A seção é identificada por um código, um título e por zero ou mais: ou outra seção, ou parágrafo, ou figura, ou tabela, ou exemplo, ou lista, ou citação, ou *link*, ou hipertexto. E assim, sucessivamente.

Uma vez armazenado o conteúdo da matéria em arquivos com extensão XML, pode-se aplicar arquivos com extensão XSL como o exemplificado abaixo, resultando em HTML.

No arquivo abaixo com extensão XSL, temos um exemplo onde mostra conteúdo com níveis de dificuldade igual a 1 (fácil).

Arquivo **01-materia.xsl**:

```
<?xml version="1.0" encoding="US-ASCII"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 version="1.0">

  <xsl:output method="html"/>

  <xsl:param name="copyright">Grupo IA-UFRGS</xsl:param>
  <xsl:param name="titulo">Ambiente de Educação a Distância</xsl:param>
  <xsl:param name="bibliografia">Referências Bibliográficas</xsl:param>
  <xsl:param name="codigo"></xsl:param>

  <xsl:template match="/">
    <xsl:apply-templates/>
  </xsl:template>

  <xsl:template match="text()">

  </xsl:template>

  <xsl:template match="materia">
    <html>
      <head>
        <title><xsl:value-of select="$titulo"/></title>
      </head>
      <body text="#000000" link="#0000ff" vlink="#0000aa" alink="#ff0000"
            topmargin="4" leftmargin="4" marginwidth="4" marginheight="4"
            bgcolor="#e9e9e9">
<table width="100%">
<tr>
<td align="center">
<a href="ServletMediador?operacao=ANTERIOR">Anterior</a>
</td>
<td align="center">
<a href="ServletMediador?operacao=PROXIMO">Próximo</a>
</td>
<td align="center">
<a href="ServletMediador?operacao=EXERCICIO">Exercício</a>
</td>
<td align="center">
<a href="ServletMediador?operacao=BIBLIOGRAFIA">Bibliografia</a>
</td>
<td align="center">
<a href="ServletMediador?operacao=USUARIO">Usuários</a>
</td>
</tr>
</table>

        <!-- THE TOP BAR (HEADER) -->
        <table width="100%" cellspacing="0" cellpadding="0" border="0">
          <tr>
            <td width="100%" height="35" valign="top" align="center"
              colspan="4" bgcolor="#00ccff">
              <h2><xsl:value-of select="titulo"/></h2></td>
          </tr>
        </table>
                    <xsl:apply-templates select="capitulo[@id=$codigo]"/>
        <!-- COPYRIGHT -->
        <br/>
        <table width="100%" border="0" cellspacing="0" cellpadding="0">
          <tr><td bgcolor="#0086b2"><img src="images/dot.gif" width="1"
              height="1"/></td></tr>
          <tr>
            <td align="center"><font size="-1" color="#0086b2"><i>
              Copyright &#169; <xsl:value-of select="$copyright"/>
            </i></font></td>
          </tr>
        </table>
      </body>
    </html>
  </xsl:template>

  <!-- CAPITULO -->
```

```
  <xsl:template match="capitulo">
    <br/>
    <h2><xsl:if test="@nivel=1"><xsl:value-of select="codigo"/>. <xsl:value-of
select="titulo"/></xsl:if></h2>
    <xsl:if test="@nivel=1"><xsl:apply-templates/></xsl:if>
  </xsl:template>

  <!-- A SECAO -->
  <xsl:template match="secao">
    <!--<h3><xsl:value-of select="@id"/>. <xsl:value-of
select="codigo"/><text> </text><xsl:value-of select="titulo"/></h3>-->
    <h3><xsl:if test="@nivel=1"><xsl:value-of select="@id"/>. <xsl:value-of
select="titulo"/></xsl:if></h3>
    <xsl:if test="@nivel=1"><xsl:apply-templates/></xsl:if>
  </xsl:template>

  <!-- A SECAO -->
  <xsl:template match="secao/secao">
    <!--<h3><xsl:value-of select="@id"/>. <xsl:value-of
select="codigo"/><text> </text><xsl:value-of select="titulo"/></h3>-->
    <h4><xsl:if test="@nivel=1"><xsl:value-of select="@id"/>. <xsl:value-of
select="titulo"/></xsl:if></h4>
    <xsl:if test="@nivel=1"><xsl:apply-templates/></xsl:if>
  </xsl:template>

  <!-- PARAGRAFO -->
  <xsl:template match="paragrafo">
    <p>
    <xsl:value-of select="."/>
    </p>
  </xsl:template>

  <!-- CITACAO -->
  <xsl:template match="citacao">
    <p>
    "<i><xsl:value-of select="."/></i>"
    </p>
  </xsl:template>

  <!-- EXEMPLO -->
  <xsl:template match="exemplo">
    <p>
    <b><xsl:value-of select="."/></b>
    </p>
  </xsl:template>

  <!-- LISTA -->
  <xsl:template match="lista">
    <li>
    <xsl:value-of select="."/>
    </li>
  </xsl:template>

  <!-- FIGURA -->
  <xsl:template match="figura">
    <img SRC="figura1.gif" BORDER="0" height="138" width="324"/>
    <br/>
    <xsl:value-of select="."/>
    <br/>
 </xsl:template>

<!-- BIBLIOGRAFIA -->
  <xsl:template match="bibliografia">
    <br/>
    <h2><xsl:value-of select="$bibliografia"/></h2>
    <table width="660" cellspacing="0" cellpadding="0" border="0">
       <tr>
         <!-- THE CONTENT PANEL -->
         <td width="500" valign="top" align="left">
           <table border="0" cellspacing="0" cellpadding="3">
              <xsl:apply-templates><xsl:sort select="@id"/></xsl:apply-
templates>
```

```
          </table>
        </td>
      </tr>
    </table>
  </xsl:template>

 <!-- REFERENCIA -->
  <xsl:template match="referencia">
    <tr><td valign="top" align="left">[<xsl:value-of
select="@id"/>]</td><td><xsl:value-of select="autor"/><text>
</text><i><xsl:value-of select="titulo"/></i>. <xsl:if test="string-
length(editor)>0"><xsl:value-of select="editor"/>. </xsl:if><xsl:value-of
select="publicacao"/>: <xsl:if test="string-length(volume)>0"><xsl:value-of
select="volume"/>, </xsl:if><xsl:if test="string-length(edicao)>0"><xsl:value-
of select="edicao"/> ed., </xsl:if><xsl:if test="string-
length(local)>0"><xsl:value-of select="local"/>. </xsl:if><xsl:if
test="string-length(mes)>0"><xsl:value-of select="mes"/>, </xsl:if><xsl:value-
of select="ano"/>. <xsl:if test="string-length(url)>0"><xsl:value-of
select="url"/></xsl:if></td></tr>
  </xsl:template>
</xsl:stylesheet>
```

Bibliografia

[ALV 97] ALVARES, Luis Otávio, SICHMAN, Jaime Simão. Introdução aos Sistemas Multiagentes. In: JORNADA DE ATUALIZAÇÃO EM INFORMÁTICA, 16., 1997, Brasília, DF. **Anais...** Brasília: UnB, 1997.

[AMD 99] ALVARES, Luis O.; MENEZES, Paulo Blauth; DEMAZEAU, Yves. Problem Decomposition: an Essencial Step for Multi-Agent Systems. In: LASKER, G. (Ed.). **Advances In Artificial Intelligence And Engineering Cybernetics**. Ontario, Canada: International Institute for Advanced Studies, 1999. v.5.

[AND 2000] ANDRADE, Adja; JAQUES, Patrícia; VICARI, Rosa; BORDINI, Rafael; JUNG, João. Uma Proposta de Modelo Computacional de Aprendizagem à Distância Baseada na Concepção Sócio-Interacionista de Vygotsky. In: WORKSHOP DE AMBIENTES DE APRENDIZAGEM BASEADOS EM AGENTES; SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, SBIE 2000, 11., 2000, Maceió, Brazil. **Anais...** Maceió: UFAL, 2000.

[AND 2001] ANDRADE, Adja; JAQUES, Patrícia; VICARI, Rosa; BORDINI, Rafael; JUNG, João. A Computational Model of Distance Learning Based on Vygotsky's Socio-Cultural Approach. In: MABLE WORKSHOP (MULTI-AGENT BASED LEARNING ENVIRONMENTS), INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE ON EDUCATION, 10., 2001, Antonio, Texas. **Proceedings...** Texas: [s.n.], 2001.

[BAQ 98] BAQUERO, B. **Vygotsky e a aprendizagem escolar**. Porto Alegre: Artes Médicas, 1998.

[BAR 99] BARROS, B.; VERDEJO, M. F. An aproach to analyse collaboration when shared strucutured workspace are used for carrying out group learning processes. In: ARTIFICIAL INTELLIGENCE IN EDUCATION, AIED, 1999, Lemans. **Proceedings...** Lemans: [s.n.], 1999.

[BER 97] BERCHT, Magda. **Avaliação Pedagógica como Fator para a Construção de Estratégias de Ensino em Ambientes de Ensino e Aprendizagem Computadorizados**. 1997. Exame de Qualificação (Doutorado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[BER 99] BERCHT, M.; MOISSA, H. E. M; VICCARI, R. M. Identificação de fatores motivacionais e afetivos em um ambiente de ensino e aprendizagem. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, SBIE, 10., 1999, Curitiba, PR. **Anais...** Curitiba: UFPR, 1999. Poster.

[BIC 99] BICA, Francine. **Eletrotutor III**: Uma abordagem multiagente para o Ensino à Distância. 1999. Dissertação (Mestrado em Ciência da

Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[BIC 2000] BICA, Francine; SILVEIRA, Ricardo A.; VICARI, Rosa M. A Multiagent Approach for Distance Learning. In: INTERNATIONAL CONFERENCE ON ENGINEERING AND COMPUTER EDUCATION**,** ICECE, 2000, São Paulo, SP. **Proceedings...** São Paulo: SENAC/SP, 2000.

[BIG 98] BIGUS, Joseph; BIGUS, Jennifer. **Constructing Intelligent Agents with Java: a programmer's guide to smarter applications**. New York: John Wiley, 1998. 379 p.

[BOR 2001] BORDINI, Rafael H.; BAZZAN, Ana L. C.; JANNONE, Rafael de O.; BASSO, Daniel M.; VICARI, Rosa M.; LESSER, Victor R. Using the DTC Scheduler to Improve Intention Selection in BDI Agents. In: MEETING FOR THE EVALUATION OF THE COOPERATIVE RESEARCH PROJECTS SUPPORTED BY CNPQ-ProTeM-CC PROGRAME, 2001, Rio de Janeiro, Brazil. **Proceedings...** Rio de Janeiro: [s.n.], 2001.

[BRA 89] BRATMAN, M. What is intention? In: COHEN P; MORGAN J; POLLACK M. (Ed.). **Intentions in Communication.** [S.1]: MIT Press, 1989.

[BUL 2000] BULL, Susan; GREER, Jim. Peer Help for Problem-Based Learning. In: INTERNATIONAL CONFERENCE ON COMPUTERS IN EDUCATION, 2000, Taipei, Taiwan. **Proceedings...** [S.l.:s.n.], 2000. v.2, p.1007-1015.

[CAN 97] CANAMERO, Dolores; VELDE, Walter Van de. Socially Emotional: Using Emotions to Ground Social Interaction. Socially Intelligent Agents. In: AAAI FALL SYMPOSIUM, 1997, Cambridge. **Proceedings...** Menlo Park: AAAI Press, 1997.

[CAS 90] CASTELFRANCHI, Cristiano. Social Power: A Point Missed in Multi-Agent, DAI and HCI. In: DEMAZEAU, Yves; MULLER, Jean-Pierre (Ed.). **Decentralized A. I.** Amsterdam, NL: Elsevier Science Publishers, 1990. p.49-62.

[CAS 97] CASTELFRANCHI, Cristiano; de ROSIS, Fiorella; FALCONE, Rino. Social Attitudes and Personalities in Agents. Socially Intelligent Agents. In: AAAI FALL SYMPOSIUM, 1997, Cambridge. **Proceedings...** Menlo Park: AAAI Press, 1997.

[CLE 78] CLERMONT, Perret; NELLY, Anne. **A Construção da Inteligência pela Interação Social**. Lisboa: Sociocultur, 1978. 366p.

[COE 94] COELHO, Helder. **Inteligência Artificial, em 25 Lições**. Lisboa: Universidade Técnica de Lisboa, Fundação Calouste Gulbenkian, 1994.

[CON 92] CONTE, Rosaria; CASTELFRANCHI, Cristiano. Mind is not Enough: Precognitive Bases of Social Interaction. In: SYMPOSIUM ON SIMULATING SOCIETIES, 1992. **Proceedings...** [S.l.:s.n.], 1992. p.93-110.

[CON 95] CONTE, Rosaria; SICHMAN, Jaime Simão. DEPNET: How to benefit from social dependence. **Journal of Mathematical Sociology**, [S.l.], v.20, n.2-3, p.161-177, 1995.

[COR 94] CORRÊA, Milton. **A Arquitetura de Diálogos entre Agentes Cognitivos Distribuídos**. 1994. Tese (Doutorado em Engenharia) – Universidade Federal do Rio de Janeiro, Rio de Janeiro.

[COR 98] CORRÊA, M.; VICCARI, R. M.; COELHO, H. Dynamics In Transition Mental Activity. In: INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS, ICMAS, 3., 1998, Paris. **Proceedings...** Los Alamitos: IEEE Computer Society, 1998.

[COR 98a] CORRÊA, Milton; COELHO, Helder. Agent's Programming from a Mental States Framework. In: BRAZILIAN SYMPOSIUM ON ARTIFICIAL INTELLIGENCE, SBIA, 14., 1998, Porto Alegre, Brazil. **Proceedings...** Berlin: Springer-Verlag, 1998.

[COR 98b] CORRÊA, Milton; COELHO, Helder. From Mental States and Architectures to Agents' Programming. In: IBERO-AMERICAN CONFERENCE ON AI. PROGRESS IN ARTIFICIAL INTELLIGENCE – IBERAMIA, 6., 1998, Lisboa. **Proceedings...** Lisboa: Springer Verlag, 1998. p.64-75.

[DAV 99] DAVID, Nuno; SICHMAN, Jaime Simão; COELHO, Helder. Extending Social Reasoning to Cope with Multiple Partner Coalitions. In: EUROPEAN WORKSHOP ON MODELLING AUTONOMOUS AGENTS IN A MULTI-AGENT WORLD, MAAMAW, 9., 1999, Valencia, Spain. **Proceedings...** Berlin: Springer-Verlag, 1999.

[DAV 2000] DAVID, Nuno; SICHMAN, Jaime Simão; COELHO, Helder. Agent-Based Social Simulation with Coalitions in Social Reasoning. In: INTERNATIONAL WORKSHOP ON MULTI AGENT BASED SIMULATION - MABS, 2., 2000, Boston, MA, USA. **Proceedings...** [S.l.:s.n.], 2000.

[DEI 2001] DEITEL, H. M.; DEITEL, P.J. **Java Como Programar**. 3.ed. Porto Alegre: Bookman, 2001.

[DEM 89] DEMAZEU, Y.; MÜLLER, J. Descentralized Artificial Intelligence. In: EUROPEAN WORKSHOP ON MODELLING AUTONOMOUS AGENTS IN A MULTI-AGENT WORLD, 1., 1989, Cambridge, England. **Proceedings...** [S.l.:s.n.], 1989.

[ECO 80] ECO, U. **Tratado geral de semiótica**. São Paulo: Perspectiva, 1980. 282p. Originalmente chamado Trattato di semiotica generale, 1976.

[FIN 93] FININ, Tim; WEBER, Jay; WIDERHOLD, Gio et al. **DRAFT Specification of the KQML Agent-Communication Language**: plus example agent policies and architectures. [S.l.]: The DARPA Knowledge sharing Initiative External Interfaces Working Group, 1993. Disponível em: <http://logic.stanford.edu/papers/README.html>. Acesso em: 29 out.2001.

[FIN 97] FININ, Tim; LABROU, Yannis; MAYFELD, James. KQML as na agent communication language. IN: BRADSHAW, Jeffrey (Ed.). **Software Agents**. Menlo Park: AAAI Press/The MIT Press, 1997. p.291-316.

[FRE 95] FREIRE, Paulo; FAGUNDES, Antônio. **Por uma Pedagogia da Pergunta**. Rio de Janeiro: Paz e Terra, 1995.

[FRO 98] FROST, Robert. **Java Agent Template (JAT)**. Disponível em: <http://java.stanford.edu/>. Acesso em: 29 out.2001.

[GIR 98] GIRAFFA, Lúcia M. M.; MÓRA, Michael da C.; VICCARI, Rosa M. Modelling the MCOE Tutor Using a Computational Model. In: BRAZILIAN SYMPOSIUM ON ARTIFICIAL INTELLIGENCE, SBIA, 14., 1998, Porto Alegre, Brazil. **Proceedings...** Berlin: Springer-Verlag, 1998.

[GIR 98a] GIRAFFA, L. M. M.; VICCARI, R. M. Tutor behaviour in a multi-agent ITS guided through mental states activities. In: INTERNATIONAL CONFERENCE ON INTELLIGENT TUTORING SYSTEMS, WORKSHOP PEDAGOGICAL AGENTS, ITS, 4., 1998, San Antonio, Texas. **Proceedings...** Berlin: Springer-Verlag, 1998.

[GIR 99] GIRAFFA, Lucia Maria Martins. **Uma arquitetura de tutor utilizando estados mentais**. 1999. Tese (Doutorado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[GIR 99a] GIRAFFA, L.M.M; VICARI, R.M. Uma arquitetura de Tutor utilizando sistemas multiagentes: da modelagem a validação pedagógica. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, SBIE, 12., 1999, Curitiba, PR. **Anais...** Curitiba: UFPR, 1999.

[GOM 99] GOMES, A. F.; VICARI, R. M. Uso de heurísticas no projeto de interfaces inteligentes. In: WORKSHOP INTERNACIONAL SOBRE EDUCAÇÃO VIRTUAL, 1999, Fortaleza, CE. **Anais...** Fortaleza: Universidade Estadual do Ceará, 1999.

[HAN 2000] HAHN, S. E; VICARI, R. M.; SILVEIRA, R. A.; MULLER, C. O; MADEIRA, M. Ensino a Distância na Internet: Uma abordagem com Inteligência Artificial. In: CONGRESSO NACIONAL DE PSICOLOGIA ESCOLAR E EDUCACIONAL, 5., 2000, Itajaí, Santa Catarina. **Anais...** Itajaí: UNIVALI, 2000.

[HÜB 2000] HÜBNER, Jomi Fred; SICHMAN, Jaime Simão. SACI: Uma Ferramenta para Implementação e Monitoração da Comunicação entre Agentes. In: IBEROAMERICAN CONFERENCE ON ARTIFICIAL INTELLIGENCE, 7.; BRAZILIAN CONFERENCE ON ARTIFICIAL INTELLIGENCE, IBERAMIA-SBIA 2000**,** 15., Atibaia, São Paulo, Brazil. **Proceedings...** São Carlos: ICM/USP, 2000.

[JAQ 98] JAQUES, Patrícia A., OLIVEIRA, Flávio M. de. Agentes de Software para Análise das Interações em um Ambiente de Ensino a Distância. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, SBIE'98, 11., 1998, Fortaleza, CE. **Anais...** Fortaleza: UFCE, 1998.

[JAQ 2000] JAQUES, Patrícia A. **Programação em Java**. Porto Alegre: Instituto de Informática – UNISINOS, 2000.

[JAQ 2001] JAQUES, Patrícia A.; KIST, Tânia; FRANZEN, Evandro; PIMENTA, Marcelo; VICARI, Rosa. Interação com Agentes Pedagógicos Animados: Um Estudo Comparativo. In: WORKSHOP SOBRE FATORES HUMANOS EM SISTEMAS COMPUTACIONAIS, IHC, 4., 2001, Florianópolis, SC. **Anais...** Florianópolis: UFSC, 2001.

[JAQ 2002] JAQUES, P. A.; ANDRADE, A. F.; JUNG, J. L.; BORDINI, R. H.; VICARI, R. M. Using Pedagogical Agents to Support Collaborative Distance Learning. In: CONFERENCE IN COMPUTER SUPPORTED COLLABORATIVE LEARNING, CSCL, 2002, Boulder, Colorado, EUA. **Proceedings...** [S.l.:s.n.], 2002.

[JEN 98] JENNINGS, Nicholas R.; SYCARA, Katia; WOOLDRIGE, Michael. **Autonomous Agents and Multi-Agent Systems - A Roadmap of Agent Research and Development**. Boston: Kluver Academic Publishers, 1998.

[JUN 2000] JUNG, João Luiz. **Uma visão Categorial dos Estados Mentais em Sistemas Multiagentes**. 2000. Trabalho Individual (Mestrado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[JUN 2001] JUNG, João; JAQUES, Patrícia; ANDRADE, Adja; BORDINI, Rafael; VICARI, Rosa. Um Agente Inteligente Baseado na Engenharia Semiótica Inserido em um Ambiente de Aprendizado à Distância. In: WORKSHOP SOBRE FATORES HUMANOS EM SISTEMAS COMPUTACIONAIS, IHC, 4., 2001, Florianópolis, SC. **Anais...** Florianópolis: UFSC, 2001. Poster.

[LEI 98] LEITE, J. C. **Modelos e Formalismos para a Engenharia Semiótica de Interfaces de Usuário**. 1998. Tese (Doutorado em Ciência da Computação) - Departamento de Informática, Pontifícia Universidade do Rio de Janeiro, Rio de Janeiro.

[LEI 99] LEITE, J. C.; de SOUZA, C. S. Uma Linguagem de Especificação para a Engenharia Semiótica de Interfaces de Usuário. In: WORKSHOP SOBRE FATORES HUMANOS EM SISTEMAS COMPUTACIONAIS, IHC, 1999, Campinas, SP. **Proceedings...** Campinas: Instituto de Computação da UNICAMP, 1999.

[LEI 2001] LEITE, Jair. Desenvolvimento de interfaces de Usuários de Sistemas Web. In: WORKSHOP SOBRE FATORES HUMANOS EM SISTEMAS COMPUTACIONAIS, IHC, 4., 2001, Florianópolis, SC. **Anais...** Florianópolis: UFSC, 2001. Minicurso. Disponível em: <http://www.dimap.ufrn.br/~jair/diuweb/index.html>. Acesso em 29 out.2001.

[LEV 99] LEVY, P. **Inteligência Coletiva. Por uma antropologia do ciberespaço**. São Paulo: Ed. Loyola, 1999.

[MAR 98] MARTINS, I. H; SOUZA, C. S. de. Uma Abordagem Semiótica na Utilização dos Recursos Visuais em Linguagens de Interface. In: WORKSHOP DE FATORES HUMANOS EM SISTEMAS COMPUTACIONAIS, IHC, 1., 1998, Maringá, PR, Brasil. **Anais...** Rio de Janeiro: PUC-RJ, 1998. p.38-47.

[MAY 96] MAYFIELD, James; LABROU, Yannis; FININ, Tim. Evaluation of KQML as an Agent Communication Language. In: WORKSHOP AGENT THEORIES, ARCHITECTURES, AND LANGUAGES, ECAI, 1995, Montreal, Canadá. **Proceedings...** Berlin: Springer-Verlag, 1996.

[MIN 85] MINSKY, M. **The Society of Mind**. London: A Tochstone Book, Simon & Schuster, 1985. p.163-172.

[MIN 86] MINSKY, M. **The Society of Mind**. New York: Simon and Schuster, 1986.

[MOR 98] MÓRA, Michael C.; LOPES, José Gabriel; VICCARI, Rosa M.; COELHO, Helder. BDI Models and Systems: Reducing the Gap. In: WORKSHOP PRE-PROCEEDINGS, INTERNATIONAL WORKSHOP ON AGENT THEORIES, ARCHITECTURES AND LANGUAGES, ATAL, 5., 1998, Paris, France. **Proceedings...** Paris: Computer Science Laboratory/University Pierre et Marie Curie, 1998.

[MOR 99] MÓRA, M. C. **Um modelo de agente executável**. 1999. Tese (Doutorado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[MOU 96] MOUSSALLE, Neila Maria. **Interações Tutor-Aluno Analisadas Através de seus Estados Mentais**. 1996. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[MOU 96a] MOUSSALLE, Neila Maria; VICCARI, Rosa Maria; CORRÊA, Milton. Intelligent Tutoring Systems Modelled Through the Mental States. In: BRAZILIAN SYMPOSIUM ON ARTIFICIAL INTELLIGENCE, SBIA, 13., Curitiba, Brazil. **Proceedings...** Berlin: Springer-Verlag, 1996.

[NOR 86] NORMAN, D. User Centered System Design. In: NORMAN, D.; DRAPER, S. (Ed.). **Cognitive Engineering**. Hillsdale, NJ: Lawrence Erlbaum, 1986. p.31-61.

[OLI 95] OLIVEIRA, F. M. Measuring Agreement and Harmony in Multi Agent Societies: A First Approach. In: BRASILIAN SYMPOSIUM ON ARTIFICIAL INTELLIGENCE, SBIA, 12., 1995, Campinas. **Proceedings...** Berlin: Springer-Verlag, 1995. p.232-241.

[ORA 99] ORACLE. **Oracle 8i JDBC Developer's Guide and Reference**. Disponível em: <http://otn.oracle.com/docs/content.html>. Acesso em: 29 out.2001.

[PAI 97] PAIVA, A. Learner Modelling for Collaborative Learning Environments. In: WORLD CONFERENCE ON ARTIFICIAL INTELLIGENCE IN EDUCATION, KNOWLEDGE AND MEDIA IN LEARNING

SYSTEMS, AI-ED, 1997, Kobe, Japan. **Proceedings...** Kobe: IOS Press, 1997.

[PAT 97] PATIL, Ramesh S.; FIKES, Richard E.; PATEL-SCHNEIDER, Peter F. et al. The DARPA Knowledge Sharing Effort: Progress Report. In: HUNS, Michael; SING, Munindar. (Ed.). **Readings in Agent**. San Francisco: Morgan Kaufmann, 1997. p.243-254.

[PEI 2000] PEIRCE, C. S. **Semiótica**. 3.ed. São Paulo: Ed. Perspectiva, 2000. (Coleção estudo, n. 46). Coleção dos manuscritos de 1931-1958.

[PIM 2000] PIMENTEL, Maria da G.; TEIXEIRA, Cesar A. C.; SANTANCHÈ, André. XML: Explorando suas Aplicações na Web. In: JORNADA DE ATUALIZAÇÃO EM INFORMÁTICA, SBC, 19., 2000, Curitiba, PR. **Anais...** Curitiba: PUCPR, 2000.

[RAO 91] RAO, Anand S.; GEORGEFF, Michael P. Modeling Rational Agents within a BDI-Architecture. In: INTERNATIONAL CONFERENCE ON PRINCIPLES OF KNOWLEDGE REPRESENTATION AND REASONING, KR&R, 2., 1991, San Mateo, CA. **Proceedings...** San Mateo: Morgan Kaufmann, 1991. p.473-484.

[RAO 95] RAO, Anand S.; GEORGEFF, Michael P. BDI agents: From theory to practice. In: LESSER, V. (Ed.). INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS, ICMAS, 1., 1995, San Francisco, USA. **Proceedings...** Menlo Park: AAAI Press, 1995. p.312-319.

[RUS 95] RUSSEL, S.; NORVIG, P. **Artificial Intelligence:** a modern approach. New Jersey: Prentice Hall, 1995.

[SHO 90] SHOHAM, Y. Nonmonotonic reasoning and causation. **Cognitive Science**, [S.l.], v.14, 1990.

[SHO 93] SHOHAM, Y. Agent-oriented programming. **Artificial Intelligence**, Amsterdam, v.60, n.1, 1993.

[SIC 92] SICHMAN, J.; DEMAZEAU, Y.; BOISSIER, O. When can Knowledge-based Systems be called Agents? In: SIMPÓSIO BRASILEIRO DE INTELIGÊNCIA ARTIFICIAL, 9., 1992, Rio de Janeiro. **Anais...** Rio de Janeiro: SBC, 1992. p.172-185.

[SIC 94] SICHMAN, Jaime Simão; DEMAZEAU, Yves; CONTE, Rosaria; CASTELFRANCHI, Cristiano. A Social Reasoning Mechanism Based On Dependence Networks. In: EUROPEAN CONFERENCE ON ARTIFICIAL INTELLIGENCE, ECAI, 11., 1994, Amsterdam. **Proceedings...** Amsterdam: John Wiley & Sons, 1994. p.188-192.

[SIC 95] SICHMAN, Jaime Simão; DEMAZEAU, Yves. Exploiting Social Reasoning to Deal with Agency Level Inconsistency. In: INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS, ICMAS, 1., 1995, San Francisco, California. **Proceedings...** Menlo Park: AAAI Press, 1995.

[SIC 98] SICHMAN, Jaime Simão. DEPINT: Dependence-Based Coalition Formation in na Open Multi-Agent Scenario. **Journal of Artificial Societies and Social Simulation**, [S.l.], v.1, n.2, 1998. Disponível em:

<http://www.soc.surrey.ac.uk/JASSS/1/2/3.html>. Acesso em: 29 out.2001.

[SIC 98a] SICHMAN, Jaime Simão; CONTE, Rosaria. On Personal and Role Mental Attitudes: A Preliminary Dependence-Based Analysis. In: BRAZILIAN SYMPOSIUM ON ARTIFICIAL INTELLIGENCE, SBIA, 14., 1998, Porto Alegre, Brazil. **Proceedings...** Berlin: Springer-Verlag, 1998.

[SIL 2000] SILVEIRA, Ricardo Azambuja. **Modelagem Orientada a Agentes Aplicada a Ambientes Inteligentes Distribuídos de Ensino - JADE - Java Agent framework for Distance learning Environments**. 2000. Tese (Doutorado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[SLO 90] SLOMAN, A. Motives mechanisms and emotions. In: BODEN, M. A. (Ed.). **The Philosophy of Artificial Intelligence**. [S.l.]: Oxford University Press, 1990.

[SOU 93] SOUZA, C. S. de. The Semiotic Engineering of User Interface Languages. **International Journal of Man-Machine Studies**, [S.l.], v.39, p.753-773, 1993.

[SUN 98] SUN SYSTEMS. **The Java Tutorial**. 1998. Disponível em: <http://java.sun.com>. Acesso em: 29 out.2001.

[SUN 98a] SUN MICROSYSTEMS. **Java Remote Method Invocation - distributed computing for Java**. 1998. Disponível em: <http://java.sun.com/marketing/collateral/javarmi.html>. Acesso em: 29 out.2001.

[SYC 98] SYCARA, Katia P. Multiagent Systems. **AI Magazine**, [S.l.], v.19, n.2, p.79-92, 1998.

[TED 98] TEDESCO, P. Mediating Meta-Cognitive Conflicts in a Collaborative Problem-Solving Situation. In: HCT WORKSHOP "BRIDGING THE GAP", 2., 1998, Brighton. **Proceedings...** Brighton: [s.n.], 1998. Abstract.

[VIC 90] VICCARI, R. M. **Um Tutor Inteligente para a programação em Lógica – Idealização, Projeto e Desenvolvimento**. 1990. Tese (Doutorado em Engenharia Eletrotécnica e Computadores), Universidade de Coimbra, Coimbra, Portugal.

[VIC 90a] VICCARI, R. M. **Inteligência Artificial:** Representação do Conhecimento. Vitória: SBC, 1990. 71p.

[VIC 92] VICCARI, R. M.; OLIVEIRA, F. M. **Sistemas Tutores Inteligentes**. 1992. Relatório de Pesquisa, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[VIC 93] VICCARI, Rosa Maria. Inteligência Artificial e Educação – Indagações Básicas. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 4., 1993, Recife, PE. **Anais...** Recife: UFPE, 1993.

[VIC 96] VICCARI, R. M.; GIRAFFA, L. M. M. Sistemas Tutores Inteligentes: abordagem tradicional x abordagem de agentes. In**:** SIMPÓSIO BRASILEIRO DE INTELIGÊNCIA ARTIFICIAL, SBIA, 13., 1996, Curitiba, PR. **Anais...** Porto Alegre: CPGCC da Ufrgs, 1996. 89p. Tutorial.

[VIC 98] VICENTE, Angel de; PAIN, Helen. Motivation Diagnosis in Intelligent Tutoring Systems. In: INTERNATIONAL CONFERENCE ON INTELLIGENT TUTORING SYSTEMS, ITS, 4., 1998, San Antônio, Texas, USA. **Proceedings...** Berlin: Springer-Verlag, 1998. p.86-95.

[VIC 2000] VICARI, R. M. The use of mental states in the construction of student model in ITS. In: INTERNATIONAL CONFERENCE ON ADVANCES IN INFRASTRUCTURE FOR ELECTRONIC BUSINESS, SCIENCE, AND EDUCATION ON THE INTERNET, SSGRR, 2000, L'Aquila, Italy. **Proceedings...** L'Aquila: Scuola Superiore Guglielmo Reiss Romoli, 2000.

[VYG 98] VYGOTSKY, L. S. **A Formação Social da Mente:** o Desenvolvimento dos Processos Psicológicos Superiores. 6.ed. São Paulo: Martins Fontes, 1998.

[VYG 98a] VYGOTSKY, L. S. **Pensamento e Linguagem**. 2.ed. São Paulo: Martins Fontes, 1998.

[WOO 95] WOOLDRIDGE, M.; JENNINGS, N. Intelligent Agents: Theory and Practice. **Knowledge Engineering Review**, [S.l.], v.10, n.2, p.115-152, 1995. Disponível em: <http://www.elec.qmw.ac.uk/dai/pubs>. Acesso em: 29 out.2001.